Fakultät II: Department für Informatik
Projektgruppe 2011/2012

# Pflichtenheft

Zur Anwendung *PlagTag*

Version 6.0

**Betreuer:**
Prof. Dr. Andreas Winter
M.Sc. Jan Jelschen

**Mitglieder:**

Tore Bierwirth
Christoph Gerken
Marion Gottschalk
Sieglinde Hahn
Maxim Klimenko
Björn Wolff
Christian Wübbeling

Oldenburg, den 30. September 2012

# Änderungsverzeichnis

### Veränderungen von Version 1.0 zu 2.0

- Überarbeitung und klarere Trennung der Zielsetzung in Kapitel 1.2, Umstrukturierung von Kapitel 1 Einleitung.
- Plagiatsdefinition und Kategorienbeschreibung als 1.5 hinzugefügt.
- Visionsanpassung in Kapitel 1.1.
- Das Vorgehensmodell der PG wurde in Kapitel 1.7 Einleitung ergänzt.
- Interviews zur Anforderungserhebung in Kapitel E Einleitung eingefügt.
- Domänenmodell und dessen Beschreibung in Kapitel 2 Allgemeine Beschreibung angepasst.
- Anwendungsfälle *Plagiatskandidat hochladen* und *Referenzdaten erheben* wurden im Anwendungsfalldiagramm in Kapitel 2 ergänzt.
- Anwendungsfälle *Plagiatskandidat hochladen* wurden gemäß dem Anwendungsfalldiagramm angepasst und bereinigt. Einige Anwendungsfälle sind (vorerst) entfallen. Fehlerfälle wurden klarer formuliert.
- Blockdiagramm in 2.1 Produkt-Einbettung hinzugefügt. Zudem Strukturanpassung und Erweiterung.
- Mockup mit detaillierter Ansicht des Strichcodes und Textvergleichsansicht in Kapitel 2.4.2 hinzugefügt.
- Die Abschnitte 2.5, 2.6 und 2.7 wurden den Anmerkungen entsprechend abgeändert.
- Anpassung der Bezeichner in Abschnitt 3.2 Anforderungsliste.
- 3.3 Leistungsanforderungen in Bezug zu Anforderungsgeber gesetzt, ansonsten entfernt. Zudem wurden die Formulierungen der Anforderungen konkretisiert.
- 3.4 Schnittstellenanforderungen in Bezug zu Anforderungsgeber gesetzt, ansonsten entfernt. Zudem wurde die Formulierung der Anforderung konkretisiert.
- Die Qualitätskriterien unter Kapitel 3.6, welche auf keine Anforderungen zurück zuführen waren, wurden entfernt.
- Das Glossar wurde überarbeitet.

### Veränderungen von Version 2.0 zu 3.0

- Die Definition des Pflichtenheftes wurde überarbeitet.
- Anpassung des Einleitungsabschnitt von Kapitel 1.
- Der Abschnitt 1.2 wurde anhand des Feedbacks angepasst.
- Überarbeitung und Erweiterung um Beispiele für Plagiate in Kapitel 1.5.
- Das Kapitel 1.6 *Abgrenzung zu anderen Anwendungen* hinzugefügt.
- Das Vorgehensmodell in Kapitel 1.7 konkretisiert. Bezug zum Spiralmodell hergestellt.
- Das Kapitel *Rechtliche Verbindlichkeit der Anforderungen* nach hinten im Pflichtenheft verschoben.

- Die Prioritäten wurden überarbeitet.
- Die Multiplizitäten im Domänenmodell in Kapitel 2 entfernt.
- Zu der Beschreibung vom Domänenmodell und dem Use Case Zwischenüberschriften eingefügt in Kapitel 2.
- Anforderungen und Abgrenzungskriterien wurden zusammengelegt und angepasst.
- Kapitel 2.4.1 und 2.4.2 wurden vereinigt.
- Zwischenüberschriften in Kapitel 2.4.1 eingefügt
- Abbildung 13, sowie den zugehörigen Text den Anmerkungen entsprechend angepasst
- Die Abschnitte 2.5 und 2.6 wurden den Anmerkungen entsprechend abgeändert.
- Funktionale Anforderungen in Kapitel 3 3.7 ergänzt.
- Funktionale Anforderungen in Anforderungsliste 3.2 geändert.
- Inhalt des Ergebnisdokuments in Kapitel 3.2 dargestellt.
- Formulierung in Leistungsanforderungen 3.3 und Schnittstellenanforderungen 3.4 angepasst.
- Überarbeitung der Anforderungen in 3.6.2 und 3.6.3.
- Überarbeitung des Inhalts im Kapitel 2.4.2.
- Glossar um Begriff SSH und Fair-Share Prinzip erweitert.
- Die Interviews in den Anhang E verschoben.
- Blockdiagramm angepasst.

**Veränderungen von Version 3.0 zu 4.0**

- Überarbeitung des Abschnittes 1.1 gemäß der Änderungswünsche.
- Entfernen der Zitiertechnik in 1.4.
- Überarbeitung, Umstrukturierung und Erweiterung um Beispiele und Schlussfolgerungen für Plagiate und Clone Types in Kapitel 1.5.
- Anforderungen aus der Visualisierung in 1.6.3 beschrieben.
- Beispiele für die Abgrenzung von Konkurrenzanwendungen eingefügt.
- Abbildung 4 bei der Beschreibung des Vorgehensmodell vertikal erstellt.
- Meilensteine in 1.7 eingefügt.
- Abschnitt 1.9: Risikoanalyse hinzugefügt.
- Den Anwendungsfall *Überprüfung durchführen* in Abbildung 11 eingefügt und beschrieben.
- Überarbeitung der Anforderungsliste.
- Anpassungen an Entwurfsanforderungen und Qualitätsanforderungen.
- Blockdiagramm angepasst (Virtuelle Maschine hinzugefügt)

- Produkt-Einbettung überarbeitet: Verweis auf virtuelle Maschine, Ansprache über SSL verschlüsselte Verbindung bei Webfrontend/Client Kommunikation
- *Überprüfung durchführen* in Abbildung 12 eingefügt, sowie *Plagiatskandidat hochladen* in *Plagiatskandidat vorverarbeiten* umbenannt.
- Tabelle 6 für *AF1.3 Überprüfung durchführen* hinzugefügt
- Diverse Umbenennungen in Kapitel 2.4.1 vorgenommen.
- Mockups: Einleitung mit Referenz auf GUI, Interview-Partner in Rollen umgewandelt, Modi vereinheitlicht, Rechtschreibkorrekturen
- Einleitungstext für Kapitel 3 überarbeitet: Rechtschreibfehler korrigiert, für noch zu spezifizierende Anforderung auf das Vorgehensmodell verwiesen.
- Konkretisierung der Formulierung in Leistungsanforderungen 3.3.
- Begriffsanpassung in Schnittstellenanforderungen 3.4.

**Veränderungen von Version 4.0 zu 5.0**

- Änderungen in Abschnitt 1.1 gemäß der Änderungswünsche.
- Anpassung des Abschnitts 1.2.
- Änderungen in Plagiate 1.5 gemäß Feedback.
- Unterschied zwischen Übersicht und Überblick in Abschnitt 1.6 deutlicher dargestellt.
- Rechtschreibfehler in Abschnitt 1.6.3 korrigiert.
- Abschnitt 1.7 überarbeitet und erweitert.
- Änderungen in Abschnitt 1.9 eingearbeitet.
- Anwendungsfälle wurden hinsichtlich der Anmerkungen überarbeitet.
- Änderungen in Produkteinbettung 2.1 gemäß Feedback.
- Beschriftung innerhalb Abschnitt 2.4.1 synchronisiert.
- Änderungen im Abschnitt 2.4.2 gemäß der Änderungswünsche.
- Die Anforderungen in Kapitel 3 wurden angepasst.
- Der Abschnitt 3.7 wurde hinzugefügt.

**Veränderungen von Version 5.0 zu 5.1**

- Glossarfunktionen eingefügt.
- Die Phasen des Vorgehensmodells 1.7 überarbeitet.
- Die Versionsinhalte für Version 2 angepasst 1.7.2.
- Die Inhalte des Ergebnisdokuments hinzugefügt 1.6.4.

**Veränderungen von Version 5.1 zu 5.2**

- Überarbeitung des Vorgehensmodells 1.7.

- Inhalte den Phasen des Vorgehensmodells 1.7 überarbeitet.
- Die Versionsinhalt für Version 3 angepasst 1.7.2.
- Den Meilenstein Endpräsentation in 1.7.2 hinzugefügt.
- Anpassung der Zuständigkeiten in 1.7.
- Seminarvortrag Deployment 1.8 eingefügt.

**Veränderungen von Version 5.2 zu 6.0**

- Korrekturen gemäß Änderungswünsche eingepflegt.
- Änderungswünsche in Abschnitt 1.1 eingefügt.
- Vorgehensmodell 1.7 inhaltlich angepasst.
- Domänenmodell in Abschnitt 2.2 an den aktuellen Stand angepasst.
- Aktivitätsdiagramme 2.4.1 angepasst.
- Seminarvortrag Deployment 1.8 erweitert.
- Gloassarbegriffe angepasst.

# Inhaltsverzeichnis


**1. Einleitung** **1**
- 1.1. Vision und Produktziel . . . 1
- 1.2. Zielsetzung dieses Dokumentes . . . 2
- 1.3. Referenzen . . . 2
- 1.4. Zielsetzung der Projektgruppe . . . 2
- 1.5. Plagiate . . . 2
  - 1.5.1. Definition und rechtliche Abgrenzung . . . 3
  - 1.5.2. Grundlegende Arten . . . 4
  - 1.5.3. Aufbauende Arten . . . 6
  - 1.5.4. Clone Types . . . 9
  - 1.5.5. Zusammenfassung . . . 12
- 1.6. Abgrenzung zu anderen Anwendungen . . . 13
  - 1.6.1. Konkurrenzanwendungen . . . 14
  - 1.6.2. Ergebnisse der Protokolle . . . 17
  - 1.6.3. Anforderungen an PlagTag . . . 18
  - 1.6.4. Inhalte des Ergebnisdokuments . . . 20
- 1.7. Vorgehensmodell . . . 20
  - 1.7.1. Projektmanagement . . . 21
  - 1.7.2. Meilensteine . . . 21
- 1.8. Deployment . . . 23
  - 1.8.1. IBM . . . 23
  - 1.8.2. Rational Unified Process . . . 24
  - 1.8.3. Harry Sneed . . . 25
  - 1.8.4. Hans-Jürgen Scheibl . . . 25
  - 1.8.5. Deutsche Industrienorm . . . 27
  - 1.8.6. Zusammenfassung und Ergebnis für die PG . . . 27
- 1.9. Risikoanalyse . . . 28
- 1.10. Überblick . . . 29

**2. Allgemeine Beschreibung** **31**
- 2.1. Produkt-Einbettung . . . 31
- 2.2. Analysemodell . . . 32
- 2.3. Allgemeine Anwendungsfälle . . . 33
- 2.4. Produkt-Funktion . . . 34
  - 2.4.1. Aktivitätsdiagramme und Anwendungsfälle . . . 35
  - 2.4.2. Mockups . . . 40
- 2.5. Anforderung an Benutzerfreundlichkeit . . . 43
- 2.6. Allgemeine Rahmenbedingungen . . . 43
- 2.7. Annahmen und Abhängigkeiten . . . 44

**3. Spezifische Anforderungen** **45**
- 3.1. Rechtliche Verbindlichkeit der Anforderungen . . . 45
- 3.2. Funktionale Anforderungen . . . 45
- 3.3. Leistungsanforderungen . . . 48
- 3.4. Entwurfs- und Schnittstellenanforderungen . . . 48
- 3.5. Datenhaltungsanforderungen . . . 49
- 3.6. Qualitätsanforderungen . . . 49
  - 3.6.1. Zuverlässigkeit . . . 49
  - 3.6.2. Sicherheit . . . 49

3.6.3. Wartbarkeit . . . . . 50
3.6.4. Erlernbarkeit . . . . . 50
3.6.5. Sonstiges . . . . . 50
3.7. Vorgehensanforderungen . . . . . 50

**A. Abbildungsverzeichnis** **52**

**B. Tabellenverzeichnis** **53**

**C. Literaturverzeichnis** **54**

**D. Glossar** **58**

**E. Anhang** **62**
E.1. Interview mit Jan Jelschen . . . . . 62
E.2. Interview mit Andreas Winter . . . . . 63
E.3. Interview mit dem Datenschutzbeauftragten . . . . . 65
E.4. Interview mit Reinhard Leidl . . . . . 66
E.5. Interview mit Andreas Winter 2 . . . . . 66

# 1. Einleitung

**Autor:** Sieglinde Hahn
Das Thema Plagiate ist seit Anfang des Jahres 2011 stärker in das Interesse der Öffentlichkeit gerückt, da Personen des öffentlichen Lebens Plagiatsvorwürfe gemacht wurden, welche sich teilweise nach Überprüfungen bestätigten. Zu diesen aufgefallenen Politikern zählen Silvana Koch-Mehrin [Den11], Jorgo Chatzimarkakis [cha11] und der Ex-Verteidigungsminister Karl-Theodor zu Guttenberg [Kla11]. Nun stellt sich die Frage warum diese Plagiate nicht schon von den Gutachtern der Dissertation erkannt wurden. Aber nicht nur in der Öffentlichkeit werden Plagiate aufgedeckt, auch in Seminararbeiten oder Abschlussarbeiten von Studenten an Universitäten kommt es immer wieder zu Plagiatsfällen. Aufgrund dieser medialen Präsenz und den Problemen an Universitäten hat sich das Thema *Entwicklung eines Frameworks zur Plagiatserkennung* der Projektgruppe (PG) *Clone Busters* ergeben.

Der Schutz des geistigen Eigentums erfordert die Suche und die Erkennung von Plagiaten. Zudem ist aus Sicht des Urheberrechts, des Patentrechts und auch der Prüfungsordnung der jeweiligen Hochschule die Einhaltung der gesetzlichen Rahmenbedingungen notwendig. Dabei werden nicht, unvollständig oder falsch zitierte Texte als Plagiat bezeichnet (vgl. Abschnitt 1.5) .

## 1.1. Vision und Produktziel

**Autor:** Sieglinde Hahn
Das Ziel des Projektes ist die Erstellung eines komponentenbasierten Frameworks zur Erkennung von Plagiaten innerhalb von Texten. Hierbei muss die Benutzeroberfläche als Webanwendung realisiert und rechenintensive Operationen nach Möglichkeit auf dem hochschuleigenen Clustersystem verteilt ausgeführt werden. Der Gedanke verschiedener, objektorientierter Komponenten muss hierbei die Verteilung der Anwendung erleichtern. Darüber hinaus muss die Erweiterbarkeit und Austauschfähigkeit einzelner Komponenten bestehen.
Zur Erkennung von Plagiaten (siehe Kapitel 1.5) muss ein Textvergleich mit Hilfe von unterschiedlichen Algorithmen durchgeführt werden. Diese müssen nicht nur wörtliche Zitate, sondern auch umformulierte Textpassagen erkennen. Die Sammlung von Referenzdaten muss mittels einer automatischen Internetrecherche ermöglicht werden. Außerdem soll die Anwendung verschiedene Arten von Quellangaben unterstützen. Dies muss die Ausgabe von korrekt zitierten Textpassagen als Plagiat vermeiden. Hierzu ist ein fünfstufiger Prozess vorgesehen. Vor diesem Prozess findet die Identifizierung von Vergleichsmaterial statt. Der Prozess ist wie folgt aufgebaut:

1. Bereitstellen mindestens eines Plagiatskandidaten durch den Anwender.
2. Bereitstellen von möglichen Referenzdokumenten durch den Anwender, was optional durchzuführen ist.
3. Automatisierte Internetrecherche mit ermittelten Schlagwörtern aus den Plagiatskandidaten, was ebenfalls optional ist.
4. Plagiatsüberprüfung mit den bereitgestellten Dokumenten und den gefundenen Internetquellen durchführen.
5. Visualisierung des Ergebnisses der Plagiatsüberprüfung und mögliche Widerlegung eines Plagiatsverdachts durch den Anwender.

Die zu vergleichenden Texte müssen in der gleichen Sprache verfasst sein, d.h., dass die Anwendung nur einen einsprachigen Vergleich zulässt. Dabei müssen die deutsche und die englische Sprache unterstützt werden. Damit unterschiedliche Textarten auf mögliche Plagiate überprüft werden können,

sollen die Datenformate PDF, HTML und TXT unterstützt werden, wobei PDFs nicht kennwortgeschützt bzw. verschlüsselt sein dürfen. Die Ergebnisse der Plagiatsüberprüfung müssen visuell, zum Beispiel in Form von Strichcodes, dargestellt werden. Die Benutzeroberfläche soll zugriffsgeschützt sein.
Als Referenz für die korrekte Funktionsweise der Anwendung wird die Dissertation von Karl-Theodor zu Guttenberg verwendet. Dabei sollen 25% aller im Wiki *GuttenPlag* [Div11] doppelt gesichteten Seiten, auf denen mit hoher Wahrscheinlichkeit mindestens eine nicht korrekt zitierte Textpassage enthalten ist, der Dissertation erkannt werden. Außerdem müssen die Referenzdokumente der zu erstellenden Anwendung zur Verfügung gestellt werden.

## 1.2. Zielsetzung dieses Dokumentes

**Autor:** Christian Wübbeling
Dieses Pflichtenheft soll die Gesamtheit der Forderungen an das Projekt aus Auftraggeber- und Auftragnehmersicht darstellen (das "Was"). Die Anforderungen aus den Anforderungserhebungen bzw. Auftraggeber-Dokumenten (dem Lastenheft) sollen hier konkretisiert werden.
Das Pflichtenheft ist für das Projekt primär die Basis für den weiteren Entwicklungsprozess, insbesondere technische Evaluationen/Machbarkeit, Entwurfs- (z.B. Architektur, technische Details) und Implementierungsphase.
Nach einer erfolgreichen Testphase erfolgt die Abnahme der Anwendung durch den Kunden, wodurch der Vertrag erfüllt ist, die eigentliche Projektlaufzeit endet und ggf. eine Wartungsphase beginnt.

## 1.3. Referenzen

**Autor:** Marion Gottschalk
In diesem Abschnitt werden zwei Bücher, auf denen die Strukturen des vorliegenden Pflichtenhefts beruht, vorgestellt. Die Struktur des Inhaltsverzeichnisses orientiert sich hauptsächlich an der beschriebenen Struktur aus dem Buch von Sommerville [KS98]. Der Unterschied zu der Struktur von Sommerville besteht darin, dass einige Abschnitte wie z.B. Mockups und *Robustheit* hinzugefügt und der Abschnitt *Definitionen, Akronyme und Abkürzungen* ausgelassen und durch ein *Glossar* ersetzt wurde. Die Bezeichnung der verschiedenen Anforderungen wurde an das Buch *Lehrbuch der Softwaretechnik - Basiskonzepte und Requirements Engineering* von Balzert [Bal09] angelehnt.

## 1.4. Zielsetzung der Projektgruppe

**Autor:** Sieglinde Hahn
Das Ziel der PG ist die Erstellung eines Frameworks zur Plagiatserkennung. Die Anwendung soll den Arbeitstitel *PlagTag* in Anlehnung an die Erkennung und Markierung von Plagiatsverdachten tragen.
Es existieren vielfältige Formen geistiger Leistung, z.B. Texte, Bilder oder Videos. Daher wird hier eine Einschränkung auf die Erkennung textueller Plagiate gesetzt.
Laut Köhler und Weber-Wulff [KWW10] gibt es viele Software-Programme zur Plagiatserkennung, trotzdem existieren Gründe, warum die Entwicklung einer eigenen Software sinnvoll ist. Hierzu gehören u.a. die unzureichende Erkennung von Plagiaten und die falsche bzw. unvollständige Zuordnung der Referenzen von korrekt zitierten Textstellen. Diese kann zur Unterstellung eines Plagiats führen, obwohl kein Plagiat vorliegt. Des Weiteren sind die vorhandenen Programme teils nicht mit dem deutschen Datenschutzrecht vereinbar.

## 1.5. Plagiate

**Autor:** Maxim Klimenko
Der Plagiatsbegriff erfordert eine verbindliche Betrachtung. Dieser ist kein direkter Bestandteil des

Urheberrechtsgesetz, daher erfolgt die Abgrenzung in *Definition und rechtliche Abgrenzung* durch einen gesetzlichen Rahmen, der das Plagiat einschränkt. Es lassen sich zudem generell zwei Plagiatskategorien identifizieren, zum einen die *grundlegenden* und zum anderen die darauf *aufbauenden Arten*. Diese lassen sich bezüglich der Anwesenheit der entsprechenden Quelle im Literaturverzeichnis und der Fußnote noch jeweils näher differenzieren, was einer Möglichkeit zur weiteren Spezifizierung von Plagiaten entspricht. Die jeweiligen Ausprägungen der Kategorien werden näher betrachtet und grob eingegrenzt. Dabei werden Beispiele falls möglich aus der Dissertation von zu Guttenberg [Gut09] aufgeführt. Dem gefolgt werden Clone Types betrachtet und in Bezug zu den Plagiatskategorien gestellt. Abschließend werden die jeweiligen *Schlussfolgerungen* aufgeführt.
Für Beispiele, falls angemessen, werden jeweils die übereinstimmenden Zeichen und die identischen Wörter angegeben. Diese Werte dienen ausschließlich als Richtwert für eine bessere Übersicht über die aufgeführten Beispiele. Dabei beinhalten die Zeichen auch Satz- und Leerzeichen, die Wörter werden durch die Satz- und Leerzeichen getrennt. Die folgenden vier Werte werden dabei erhoben um die Übereinstimmung zu bewerten:

1. Zeichen insgesamt: Alle Zeichen des originalen Ausschnitts.
2. Zeichen übereinstimmend: Alle mit dem Original übereinstimmenden Zeichenketten in erkennbarer Folge z.B. sind zwischen „Hund“ und „Hunde“ vier Zeichen übereinstimmend. Dieser Wert wird über alle Übereinstimmungen aufsummiert. Hier gilt die Bedingung, dass die Menge der Zeichenketten des Plagiats eine Untermenge der Mengen der Zeichenketten des Originals sein müssen.
3. Wörter insgesamt: Alle Wörter des originalen Ausschnitts.
4. Wörter identisch: Alle mit dem Original identischen Wörter in erkennbarer Folge z.B. wäre „Hund“ nicht identisch mit „Hunde“. Hier gilt die Bedingung, dass die Menge der Wörter des Plagiats eine Untermenge der Menge der Wörter des Originals sein müssen.

### 1.5.1. Definition und rechtliche Abgrenzung

Der Begriff des Plagiats wird von Dreyer [GDM09] als die beabsichtigte Aneignung einer fremden Leistung, um diese als eigene auszugeben, beschrieben. Da es im Urheberrechtsgesetz (UrhG) keine explizite Definition für Plagiate, aber ein geltendes Rahmenwerk gibt, werden hier die entsprechend verknüpften Bedingungen im Kontext von wissenschaftlichen Arbeiten näher betrachtet. Dies dient der Abgrenzung des Plagiatsbegriff aus einer rechtlich orientierten Perspektive.
Die folgenden Definitionen und rechtlichen Abgrenzungen zum Plagiat, Eigenplagiat, Doppelschöpfung und zur unbewussten Entlehnung basieren auf den Ausführungen von Dreyer [GDM09].

#### Definition

Es gibt das Recht auf Schutz der geistigen geschriebenen Leistung (§2 UrhG). Diese Leistung, falls veröffentlicht, darf verwendet werden, um wissenschaftliche Arbeiten zu verfassen, muss dazu aber zitiert werden (§51 UrhG). Wird eine solche Leistung übernommen und als eigene ausgegeben, so ist es ein Verstoß gegen §23 UrhG und kann, bei Absicht, als Plagiat bezeichnet werden. Dabei ist die *freie Benutzung* (§24 UrhG), bzw. die Parodie, vom Plagiat abzugrenzen. Im Fall des Plagiats können jedoch Ansprüche geltend gemacht werden. Zudem wird in §97 UrhG zwischen einer absichtlichen oder einer unabsichtlichen Aneignung nicht differenziert.
Für übernommene Leistungen gilt daher grundsätzlich die inhaltlich korrekte Quellenangabe (§63 UrhG). Eine formell inkorrekte Quellenangabe deutet aber nicht direkt auf ein Plagiat hin, solange deutlich wird, dass der Inhalt übernommen wurde und die Absicht, diese als eigene auszugeben, nicht bestand. Somit ist die Absicht, eine fremde Leistung als eigene auszugeben, ein Bestandteil der Begriffsdefinition. Bei einer deutlichen Angabe darüber, dass es keine eigene Leistung ist, ist auch keine Absicht zu erkennen. Eine formell korrekte Quellenangabe kann ebenso kein Plagiat ausschließen, denn im Fall eines Zitats von einem Zitat wird das Urheberrecht eines Dritten verletzt.

#### Abgrenzung zur Parodie

Laut Schwartmann [Sch08] entspricht die freie Benutzung der Parodie, welche bestimmte Kriterien erfüllen muss, um keine Genehmigung des Autors zu erfordern. So benötigt die Parodie u.a. einen Bezug zum Original und muss zugleich einer vollständig neuen Leistung entsprechen.

#### Abgrenzung zum Eigenplagiat

Die Form des Eigenplagiates stellt keinen Verstoß gegen das Urheberrechtsgesetz dar, denn das Urheberrecht liegt weiterhin bei demselben Autor.

#### Abgrenzung zur Doppelschöpfung

Eine weitere Variante ist die Doppelschöpfung. Diese gilt, falls zwei Autoren vollkommen unabhängig voneinander eine identische Leistung erbracht haben. In diesem Fall haben auch beide Autoren das Urheberrecht an der Leistung.

#### Abgrenzung zur unbewussten Entlehnung

Die Form der unbewussten Entlehnung beschreibt den subjektiv empfundenen Anspruch auf eine Leistung, z.B. weil diese gelesen und nach einiger Zeit als eigene empfunden wurde.

#### Schlussfolgerung

Insgesamt haben Plagiate im engeren Sinne drei wesentliche Bestandteile: Die Absicht, die Aneignung einer fremden Leistung und die Art der Verwendung. Für die Anwendung von §97 UrhG ist die Absicht aber nicht relevant. Für Eigenplagiate gilt zwar §97 UrhG nicht direkt, jedoch können weitere vertragliche Rahmenbedingungen existieren, welche diesen Aspekt regeln, z.B durch das Arbeitsverhältnis. Daher verdienen die Eigenplagiate auch eine entsprechende Beachtung. Plagiate im weitesten Sinne wären entsprechend jene, die von §97 UrhG und sonstigen vertraglichen Bedingungen erfasst werden. Die formelle Korrektheit der Quellenangabe wird dabei getrennt von der Absicht der Leistungsaneignung betrachtet. Es ist kein Plagiat im engeren Sinne, solange eine Quelle deutlich zu erkennen ist. Im weiteren Sinne ist jedoch eine korrekte Quellenangabe notwendig. Die Doppelschöpfung entspricht einer zufälligen Leistungsübereinstimmung von zwei Autoren, was als ein spezieller Sonderfall betrachtet werden kann, ebenso die unbewusste Entlehnung. Eine Konsequenz ist, dass eine automatisierte Prüfung dieser beiden Fälle auf Grund des Bedarfs von externen rechtlichen Entscheidungen und Informationen, falls überhaupt, nur mit hohem Aufwand realisierbar wäre.
Es erfolgt im Folgenden eine Einschränkung auf Plagiate im weiteren Sinne, da diese aus einer rechtlich orientierten Perspektive relevant sind. Es können auch für Eigenplagiate rechtliche Einschränkungen existieren, sodass diese ebenfalls beachtet werden sollen. Die beiden Sonderfälle der Doppelschöpfung und der unbewussten Entlehnung werden auf Grund der mangelnden Realisierbarkeit ausgeschlossen. Ebenso die Parodie, welche in wissenschaftlichen Arbeiten unüblich ist.

### 1.5.2. Grundlegende Arten

Die zwei grundlegenden Arten von Plagiaten sind die Komplettplagiate und die Verschleierung. Je nach Umfang der wörtlichen Übereinstimmung lassen sich diese differenzieren. Die folgenden Plagiatsarten wurden GuttenPlag [gut11] entnommen.

#### Komplettplagiat

Ein Komplettplagiat liegt bei einer nahezu wörtlichen Leistungsübernahme vor, wobei dies nicht entsprechend als direktes Zitat gekennzeichnet ist. Für den Grenzwert von der wörtlichen Übereinstimmung zur Verschleierung existiert noch kein Erfahrungswert.

Solch ein, von GuttenPlag [gut12b] gefundenes, Plagiat liegt z.B. auf Seite 48 der Dissertation von zu Guttenberg [Gut09] vor:

> *Die verfassungsrechtlich gewollte „Langsamkeit“ der Politikprozesse in den USA ist in den vergangenen Jahrzehnten häufg durch das Phänomen des „divided government“ verstärkt worden.*

Zum Vergleich das nicht referenzierte, aber wörtlich übernommene, Original von Wasser [Was97]:

> *Die verfassungsrechtlich gewollte „Langsamkeit“ der Politikprozesse in den USA ist in den vergangenen Jahrzehnten häufig durch das Phänomen des divided government verstärkt worden.*

Der Text wurde fast vollständig übernommen. So sind von 182 Zeichen 180 übernommen worden (ca. 99%) und alle 23 Wörter sind identisch (100%).

**Verschleierung**

Die Form der Verschleierung findet bei umformulierten und veränderten Texten Anwendung, falls diese nicht mit einem indirekten Zitat deutlich gemacht wurden. Für den Grenzwert von der wörtlichen Übereinstimmung zum Komplettplagiat existiert noch kein Erfahrungswert.

Es ist u.a. auf Seite 71 der Dissertation von zu Guttenberg [Gut09] eine Verschleierung von GuttenPlag [gut12c] gefunden worden:

> *A. Peters charakterisiert den Verfassungsentwurf 1984 wie auch den aus dem Jahre 1994) als „in dem Sinne revolutionär“, als er sich als normativ diskontinuierlich zum geltenden Verfassungsrecht auffasste. [169]*

Im Vergleich das als indirektes Zitat gekennzeichnete Original von Fuchs et al. [FHP02]:

> *[13] Anne Peters, a.a.O. (Anm.8), S.492 ff. charakterisiert den Verfassungsentwurf 1984 wie auch den aus dem Jahre 1994 daher auch als „in dem Sinne revolutionär, als sie sich als normativ diskontinuierlich zum geltenden Verfassungsrecht auffassten.“*

Der Text ist z.T. geändert worden, aber es besteht dennoch eine hohe Ähnlichkeit. Es sind von 250 Zeichen 201 übernommen worden (ca. 80%), zudem sind von 36 Wörtern 27 identisch (75%). Wörtlich übernommenes hätte entsprechend gekennzeichnet werden müssen. Wäre der Text vollständig umformuliert, so wäre es auch kein Plagiat gewesen. Falls aber zusätzlich der Text nicht referenziert gewesen wäre, würde es wiederum als Verschleierung bezeichnet.

**Schlussfolgerung**

Basierend auf diesen beiden Arten lassen sich verschiedene Ausprägungen der möglichen Plagiatsstrukturen, z.B durch Umfang und Anzahl der Plagiate, identifizieren. Diese werden hier als aufbauende Arten bezeichnet. Aus den grundlegenden Arten resultiert ein Kriterium zur Unterscheidung: Die wörtliche Übereinstimmung. Bei einer hohen wörtlichen Übereinstimmung handelt es sich um ein Komplettplagiat. Ist die wörtliche Übereinstimmung gering, also der Text umformuliert, so ist es eine Verschleierung. Dabei existiert hier noch kein Erfahrungswert für die konkrete Abgrenzung. Es besteht die Chance durch die Erkennung von Komplettplagiaten, via Parametrisierung der wörtlichen Übereinstimmung, auch zugleich ein Teil der Verschleierungen zu entdecken. Der Grund liegt in der Definition für Verschleierung, denn diese erlaubt wörtliche Übereinstimmungen bis zu einem bestimmten Umfang. Verschleierungen mit einer Übereinstimmung von 0% würden hierbei nicht erkannt werden können. Hieraus resultiert insgesamt eine Priorisierung zu Gunsten der Komplettplagiate, wobei beide Arten relevant sind.

### 1.5.3. Aufbauende Arten

In Abhängigkeit zur Verwendung der Quellenangabe und -urheber, dem Umfang und der Anzahl der Wortübereinstimmungen lassen sich sechs weitere Arten bestimmen, wobei diese auf die zwei grundlegenden Arten zurückzuführen sind. Die folgenden Plagiatsarten wurden aus GuttenPlag [gut11] entnommen.

#### Alibi-Fußnote

Laut Schimmel [Sch11] werden bei dieser Form „[...] zum Kaschieren flächiger Plagiate punktuelle Belege gesetzt“. Implizit führt diese Art zu Komplettplagiaten und Verschleierungen. GuttenPlag [gut11] differenziert die Alibi-Fußnote noch weiter, zum einen das *Bauernopfer* und zum anderen das *verschärfte Bauernopfer*. Beim Bauernopfer wird vorgegeben einzelne Passagen übernommen zu haben. Im Vergleich wird beim verschärften Bauernopfer vorgegeben eine Leistung selbstständig erschaffen zu haben. Dies soll durch eine Referenz auf eine weitere Leistung, z.B als Vergleich, bestätigt werden.

Ein Bauernopfer wurde auf Seite 149 der Dissertation von zu Guttenberg [Gut09] von GuttenPlag [gut12e] gefunden:

> *Bezeichnet man nun den souveränen Staat als Rechtsvoraussetzungsbegriff und die Konstituente, den „pouvoir constituant“, als nur aus diesem Grund frei, besteht nach dem so genannten „normativen staatsbezogenen Verfassungsbegriff“ folglich eine Konnexität von souveränem Staat und Verfassung[FN 416] [...].*

Im Vergleich das Original des referenzierten Begriffs von Blumenwitz [Blu03], woraus mehr entnommen wurde als lediglich nur der eine Begriff:

> *Der souveräne Staat ist Rechtsvoraussetzungsbegriff und die Konstituente, der „pouvoir constituant“, nur aus diesem Grund frei.[FN 5] Folglich besteht eine Konnexität von souveränem Staat und Verfassung (sog. normativer staatsbezogener Verfassungsbegriff).*

Insgesamt stimmen 240 von insgesamt 256 Zeichen überein (ca. 94%) und 26 Wörter von insgesamt 31 sind identisch (ca. 84%). Es ist aus dem Text von Blumenwitz [Blu03] erkennbar mehr entnommen worden als nur „normativen staatsbezogenen Verfassungsbegriff“ [Gut09].

Ein verschärftes Bauernopfer ist auf Seite 45 der Dissertation von zu Guttenberg [Gut09] von GuttenPlag [gut12a] gefunden worden:

> *[Fn 90]In den mehr als zweihundert Jahren der amerikanischen Verfassung, hat Deutschland das Ende des Heiligen Römischen Reichs gesehen, den Rheinbund, den Deutschen Bund, 1848, später den Norddeutschen Bund, die Bismarck'sche Reichsverfassung von 1871, die Weimarer Verfasssung, die Rechtlosigkeit und Willkürherrschaft des Dritten Reichs, die Besatzungszeit, zwei Verfassungen der DDR und das Grundgesetz. Vgl. auch zu dieser Gegenüberstellung G. Casper, Die Karlsruher Republik, Rede beim Staatsakt zur Feier des fünfzigjährigen Bestehens des Bundesverfassungsgerichts am 28. September 2001 in Karlsruhe, http://www.bverfg.de/texte//deutsch/aktuell/Casper.html*

Der vollständige originale Text von Casper [Cas01] des angeblichen Vergleichs, welcher deutlich übernommen wurde:

> *In den mehr als zweihundert Jahren der selten förmlich geänderten amerikanischen Verfassung hat Deutschland das Ende des Heiligen Römischen Reichs gesehen, den Rheinbund, den Deutschen Bund, 1848, später den Norddeutschen Bund, die bismarcksche Reichsverfassung von 1871, die Weimarer Verfasssung, die Rechtlosigkeit und Willkürherrschaft des Dritten Reichs, die Besatzungszeit, zwei Verfassungen der DDR und das Grundgesetz.*

Insgesamt stimmen 398 von insgesamt 425 Zeichen überein (ca. 94%) und 51 Wörter von insgesamt 55 sind identisch (ca. 93%). Da der Tippfehler in „Verfasssung“ ebenfalls von zu Guttenberg [Gut09] übernommen wurde, wird dies ebenfalls als ein identisches Wort gezählt. Der Satz „Vgl. auch zu dieser Gegenüberstellung G. Casper [...]“ von zu Guttenberg bedeutet nicht, dass der Text nahezu wörtlich übernommen worden ist.

### Halbsatzflickerei

Zur Halbsatzflickerei zählt die Übernahme einzelner Wörter und Satzfragmente. Somit gibt es bei dieser Kategorie mehrere Komplettplagiate.

Ein solches Plagiat wurde z.B. auf der Seite 77 der Dissertation von zu Guttenberg [Gut09] von GuttenPlag [gut12d] gefunden:

> *[...] die Deutschen lehnten den drohenden Verlust der DM zugunsten eines ECU ab, Großbritannien sträubte sich gegen eine gemeinsame Sozialpolitik. Spanien, Portugal und Griechenland erwarteten mehr Geld aus dem Kohäsionsfonds, Dänemark plante sich von der gemeinsamen Außenpolitik fernzuhalten, Deutschland und die Benelux-Staaten begrüßten allerdings den weiteren Schritt hin zu engerer Zusammenarbeit. Die Kontroversen spiegelten sich in drei Volksbefragungen wider.*

Zum Vergleich der nicht referenzierte originale Text, von de Beer [Bee00], welcher wie ein „Bausatz“ benutzt wurde:

> *Die Deutschen lehnten den drohenden Verlust der DM zugunsten des ECU ab, die Briten sträubten sich gegen eine gemeinsame Sozialpolitik, Spanier, Portugiesen und Griechen erwarteten mehr Geld aus dem Kohäsionsfonds. Die Kontroversen spiegelten sich auch in drei Volksbefragungen wieder.*

Insgesamt stimmen 263 von insgesamt 285 Zeichen überein (ca. 92%) und 31 Wörter von insgesamt 39 sind identisch (ca. 79%). Zu Guttenberg hat lediglich eine alternative Länderbezeichnung gewählt und einen Satz ‚von „[...] Dänemark plante [...]“ bis „[...] engerer Zusammenarbeit“, eingeschoben. Insgesamt mit kleineren Variationen, aber immernoch erkennbar derselbe Text.

### Shake & Paste

Ähnlich zur Halbsatzflickerei, bezieht das Shake and Paste aber ganze Sätze und Absätze verschiedener Autoren mit ein. Diese Textfragmente bilden insgesamt ein Mosaik verschiedener Leistungsurheber.

Ein Shake and Paste wurde z.B. auf der Seite 330 der Dissertation von zu Guttenberg [Gut09] von GuttenPlag [gut12g] gefunden:

> *[951] Dazu aus der Lit.: H. Lecheler, Das Subsidiaritätsprinzip, 1993; P. Häberle, Das Prinzip der Subsidiarität aus der Sicht der vergleichenden Verfassungslehre, in: AöR 119 (1994), S. 169 ff.; M Zuleeg, Das Subsidiaritätsprinzip im Europarecht, in: Mélanges en hommage à F. Schockweiler, 1999, S. 635 ff.*

Im Vergleich dazu der originale Text von Häberle [Hä06], welcher verschiedene Urheber referenziert und wie ein „Bausatz“ benutzt wurde:

> *[274] Dazu aus der Lit.: M. Zuleeg, Das Subsidiaritätsprinzip im Europarecht, in: Mélanges en hommage à F. Schockweiler, 1999, S. 635 ff.; [...]; H. Lecheler, Das Subsidiaritätsprinzip, 1993; P. Häberle, Das Prinzip der Subsidiarität aus der Sicht der vergleichenden Verfassungslehre, AöR 119 (1994), S. 169 ff.*

Insgesamt stimmen 298 von insgesamt 311 Zeichen überein (ca. 94%) und 44 Wörter von insgesamt 46 sind identisch (ca. 93%). Die zwei Verweise auf die Literatur von Häberle [Hä06], inklusive „Dazu aus der Lit.:“, sind wörtlich übernommen worden. Die verschiedene Literatur erfüllt hier die Bedingung der verschiedenen Urheber.

### Strukturplagiat

Das Strukturplagiat liegt bei einer übereinstimmenden Struktur vor, dies bedeutet z.B. im Inhaltsverzeichnis oder aber auch innerhalb von Aufzählungen/-listungen. Das Strukturplagiat ist nicht als solches zu werten, falls eine Parodie vorliegt, welche eine entsprechende Struktur für eine vollständig neue Leistung erfordern kann. Der Fall der Parodie ist in wissenschaftlichen Arbeiten unüblich.

Ein Strukturplagiat ist in der Gliederung der Dissertation von zu Guttenberg [Gut09] von GuttenPlag [gut12h] gefunden worden:

> *(3) Leitbilder und europäische Ideale in der politischen Auseinandersetzung. . . [...] 102*
> *(a) Das Ideal einer „Föderation von Nationalstaaten“ . . . 103*
> *(b) Das Ideal eines „Europas der Nationen“ . . . [...] 106*
> *(c) Das Ideal eines „Europas der Regionen“ . . . [...] 108*
> *(d) Ein offenes Leitbild mit Gemeinschaftsansatz . . . [...] 109*
> *(e) Zwischenfazit . . . [...] 110*

Zum Vergleich die originale Gliederung, welche nahezu wörtlich und strukturell von Volkmann-Schluck [VS01] übernommen wurde:

> *4. Leitbilder in der Debatte...[...] 22*
> *4.1. Föderation von Nationalstaaten...[...] 23*
> *4.2. Europa der Nationen...[...] 25*
> *4.3. Europa der Regionen...[...] 26*
> *4.4. Offenes Leitbild mit Gemeinschaftsansatz...[...] 27*
> *4.5. Wirtschaftliche Supermacht statt Superstaat...[...] 28*
> *4.6. Zwischenfazit...[...] 30*

Insgesamt stimmen 136 von insgesamt 195 Zeichen überein (ca. 70%) und 13 Wörter von insgesamt 22 sind identisch (ca. 60%). Diese Werte sind exklusive aufeinander folgender Punkte, Kapitel- und Seitennummerierungen. Hier liegt eine erkennbare strukturelle Übereinstimmung vor.

### Kopiertes Zitat

Wird von einem Autor eine Leistung zitiert und von diesem Zitat fertigt ein dritter Autor wiederum ein Zitat an, referenziert aber nicht den Autor des Originals, so zählt es als Kopiertes Zitat. Hierbei liegt eine formell inkorrekte Quellenangabe vor, die inhaltliche Ausprägung bezieht sich nicht auf den Urheber der Leistung.

Ein Kopiertes Zitat, aber kein Plagiat, wurde auf der Seite 205 der Dissertation von zu Guttenberg [Gut09] von GuttenPlag [gut12f] gefunden, dieses liegt zwischen Zeile 16 und 19. Hier wurde eine Rede von Kennedy [Ken62] korrekt Zitiert. Die Zeilen 1-28 basieren dabei insgesamt auf einem zusammenhängenden Text des Autors Weege [Wee05], welcher nicht referenziert wurde. Dieses Beipsiel enthält nur das kopierte Zitat (Zeile 16-19) und nicht den gesamten Text (Zeilen 1-28), welcher ebenfalls weitere Plagiate enthält. Die von zu Guttenberg [Gut09] zitierte Rede von Kennedy [Ken62], welche auch von Weege [Wee05] wörtlich übernommen wurde:

> *„Die Vereinigten Staaten sehen auf dieses große neue Unterfangen mit Hoffnung und Bewunderung. Wir betrachten ein starkes und vereintes Europa nicht als Rivalen, sondern als Partner. Seinen Fortschritt zu unterstützen, war siebzehn Jahre lang das Hauptanliegen unserer Außenpolitik.“ [579]*

Diese Rede ist zwar korrekt zitiert, jedoch befindet sich die Rede an derselben Stelle im Text, wie auch bei Weege. Daher wurde das Zitat von Weege plagiiert ohne dies entsprechend zu referenzieren. Dies resultiert in ein Strukturplagiat mit einem, hier korrektem, kopierten Zitat. Wäre alternativ insgesamt nur Weege referenziert gewesen statt Kennedy, so wäre die Rede ein Kopiertes Zitat ohne Strukturplagiat.

#### Eigenplagiat

Aus der Perspektive des Urheberrechtsgesetz stellen Eigenplagiate zwar kein Problem, im Sinne von §97 UrhG, dar. Jedoch können weitere vertragliche Rahmenbedingungen existieren, welche diese Art des Plagiats mit entsprechenden Konsequenzen einbeziehen. Hierfür gibt es kein Beispiel aus der Dissertation von zu Guttenberg. Als Beispiel für ein Eigenplagiat wird daher der Züricher VWL Professor Bruno Frey gewählt. Dieser hat in verschiedenen Publikationen eigene Arbeiten plagiiert.

Aus der Sammlung von Freys Eigenplagiaten [fre12] ist hier ein Beispiel der Publikation aus der *European Journal of Law and Economics* [Fre05] von 2005:

> *Many economic scholars are likely to be in complete disagreement with this interpretation of the behavior of the referees. They like to think that the referees only ask for changes improving the paper in the interests of the author, but refrain from interfering any further.*

Im Vergleich dazu, ebenfalls von Frey, der originale Text aus dem *Public Choice* [Fre03] von 2003, also zwei Jahre zuvor:

> *Many economic scholars are likely to be in complete disagreement with this interpretation of the behavior of the referees. They like to think that the referees only ask for changes improving the paper in the interests of the author, but refrain from interfering any further.*

Insgesamt stimmen 274 von insgesamt 274 Zeichen überein (100%) und 45 Wörter von insgesamt 45 sind identisch (100%). Ungeachtet der Konsequenz für Frey ist dies ein Beispiel für ein Eigenplagiat mit einer Übereinstimmung von 100%.

#### Schlussfolgerung

Plagiate lassen sich in zwei Kategorien unterscheiden: zum einen die grundlegenden Arten und zum anderen die aufbauenden Arten. Die grundlegenden Arten sind abhängig vom Umfang der wörtlichen Übereinstimmung, die aufbauenden Arten von Quellenangabe und von der Häufigkeit der einzelnen Plagiate.
Bevor die Bestimmung der aufbauenden Arten möglich ist, müssen zuvor die jeweiligen grundlegenden Arten identifiziert worden sein. Somit ist es erforderlich zuerst die grundlegenden Arten zu realisieren. Anschließend können alle für das Projekt zielführenden Arten realisiert werden. Dies bedeutet alle Arten, wie sie auch von GuttenPlag in der zu Guttenberg Dissertation gefunden wurden: Halbsatzflickerei, Shake and Paste, Alibi-Fußnote und Strukturplagiat. Die Reihenfolge ergibt sich aus dem Verhältnis der erwarteten Komplexität. Erst danach sollten Eigenplagiate und kopierte Zitate realisiert werden, da diese die Messung des Projekterfolgs nicht beeinflussen.

### 1.5.4. Clone Types

Eine alternative Kategorisierung von Plagiaten sind die vier verschiedenen *Clone Types*. Diese werden bei der Erkennung von Plagiaten in Quelltexten verwendet. Ein Clone muss dabei keiner rechtlichen Verletzung entsprechen. Die Definitionen der ersten drei Typen basieren auf den Aussagen von Mens und Demeyer [MD08], die Erweiterung um den vierten Typ basiert auf Chanchal et al. [RCK09].

Als Basis für die Beispiele der verschiedenen Clone Types wird die folgende Java-Methode verwendet, somit erfolgt jeder Vergleich direkt in Bezug zu dieser Methode:

```
public int quad (int x){
        return x*x;
}
```

Die verschiedenen Beispiele werden auch auf die natürliche Sprache angewendet. Hierfür wird als Basis das folgende Sprichwort verwendet:

> *Die dümmsten Bauern haben die dicksten Kartoffeln.*

### Typ-1

Bei einer vollständig wörtlichen Übereinstimmung wird diese als ein *Typ-1 Clone* bezeichnet. Hierbei werden Formatierung und Kommentare nicht beachtet.

Als Beispiel für einen Typ-1 Clone sei folgende Methode gegeben:

```
public int quad (int x){
        //calculate and return
        return x*x;
}
```

Bis auf den Kommentar sind diese Methoden absolut identisch. Aus Sicht des Typ-1 Clones sind diese sogar zu 100% identisch. Kommentare in diesem Sinne gibt es allerdings keine in natürlichen Sprachen. Insgesamt entspricht dieser Typ den Komplettplagiaten, somit gilt auch hier das entsprechende Beispiel aus 1.5.2.

Unter der Prämisse, dass ein Kommentar als sprachliches Mittel ohne direkten Einfluss auf die Funktionalität betrachtet wird, könnten z.B. die verschiedenen Artikel der deutschen Sprache („der“, „die“, „das“) ebenfalls als solche betrachtet werden. Denn der Artikel geht implizit aus dem jeweiligen Nomen hervor:

> *dümmsten Bauern haben dicksten Kartoffeln.*

Es existiert noch keine Erfahrung dafür wie praktikabel der Ausschluss der Artikel für die Plagiatserkennung ist. Bei dem hier eröffneten Beispiel bleibt die Aussage des Satzes, also die Funktionalität, identisch.

### Typ-2

Werden lediglich Wörter, bzw. Bezeichner, ausgetauscht, aber die Syntax-Struktur bleibt identisch, so ist es ein *Typ-2 Clone*.

Als Beispiel hier eine Methode mit lediglich einem anderen Bezeichner für die Variable.

```
public int quad (int kappa){
        return kappa*kappa;
}
```

Diese Methoden haben eine zu 100% identische Syntax-Struktur. Für natürliche Sprachen ist dies ebenfalls mit Synonymen möglich. Dies kann als ein spezieller Fall der Verschleierung betrachtet werden. Im Rahmen der Recherchen konnte kein adäquates Beispiel in der Dissertation von zu Guttenberg gefunden werden.

Für das hier eröffnete Beispiel der natürlichen Sprache kann der folgende Satz als Typ-2 Clone betrachtet werden:

> *Die dümmsten Agronomen haben die dicksten Solaneen.*

Die syntaktische Struktur und die Aussage sind identisch geblieben, wobei zwei Wörter ausgetauscht wurden. Diese Form entspricht einer Verschleierung.

### Typ-3

Wurden Anweisungen verändert, hinzugefügt oder entfernt, so ist es ein *Typ-3 Clone*.

Als Beispiel ein Typ-3 Clone mit einer zusätzlichen Anweisung:

```
public int quad (int x){
        counter+=x;
        return x*x;
}
```

Die zusätzliche Anweisung verändert hier weder die Funktionalität noch die Syntax-Struktur der unangetasteten Zeilen. Insgesamt entspricht dieser Typ den Verschleierungen und Halbsatzflickereien. Somit gilt hier z.B. auch das entsprechende Beispiel für Halbsatzflickereien aus 1.5.2.

Für das hier eröffnete Beispiel der natürlichen Sprache kann der folgende Satz als Typ-3 Clone betrachtet werden:

> *Die dümmsten Bauern, egal wo, haben die dicksten Kartoffeln.*

Der Satz ist um einen eingeschobenen Nebensatz erweitert worden ohne dessen Aussage zu verfälschen.

### Typ-4

Bei einer identischen Funktion, aber einer unterschiedlichen Ausprägung, wörtlich wie auch syntaktisch, erfolgt die Kategorisierung in einen *Typ-4 Clone*.

Hier als Beispiel eine Methode mit einer identischen Funktionalität:

```
public int quadRa (int kappa){
        int x = -1*kappa;
        int y = x*x;
        kappa = y;
        return kappa;
}
```

Die syntaktische Struktur hat sich verändert, ebenso wie die Bezeichnung der Methode und der Variablen. Im Rahmen der Recherchen konnte kein adäquates Beispiel in der zu Guttenberg Dissertation gefunden werden.

Für das hier eröffnete Beispiel der natürlichen Sprache kann der folgende Satz als Typ-4 Clone betrachtet werden:

> *Das Volumen der Solaneen ist reziprok proportional zu der zerebralen Kapazität des Agronomen.*

Es stimmt nicht ein Wort überein und die syntaktische Struktur ist ebenfalls anders, die Aussage ist aber nahezu identisch geblieben. Dies kann ebenfalls als Beispiel für eine Verschleierung mit einer wörtlichen Übereinstimmung von 0% betrachtet werden.

### Schlussfolgerung

Die Clone Types lassen sich exemplarisch in Bezug, zu der vorhergehenden Kategorisierung, stellen. So entspricht der Typ-1 Clone dem Komplettplagiat. Für Typ-2 Clones, kann eine Spezifizierung von Verschleierung Entsprechung finden, falls nur die Wörter ausgetauscht wurden, aber nicht deren

grammatikalische Abfolge. Die Typ-3 Clones entsprechen, je nach konkreter Ausprägung, einer Verschleierung oder einer Halbsatzflickerei. Da die Verschleierung auch bei vollkommen umformulierten Texten greift, kann dieser auch der Typ-4 Clone zugerechnet werden.
Insgesamt sind die verschiedenen Clone Types spezifische Fälle der Komplettplagiat und Verschleierungen, somit alternative aufbauende Arten. Daher bleibt es erforderlich zuerst die grundlegenden Arten zu erkennen um diese dann den Clone Types zuzuordnen. Algorithmen zur Erkennung der Clone-Types können auch für die natürliche Sprache verwendet werden, in welchem Umfang und unter welchen Bedingungen muss noch geprüft werden.

### 1.5.5. Zusammenfassung

Die folgende Auflistung enthält die verschiedenen Betrachtungen des Plagiats und der Kategorien. Diese ist insgesamt geordnet nach Einfluss, bzw. Relevanz, zur Erreichung des Projektziels, beginnend mit der höchsten Relevanz. Bei gleicher Relevanz wird nach erwarteter Komplexität für die Realisierung sortiert, beginnend mit der geringsten Komplexität. Dabei ist die Erkennung von Komplettplagiaten als besonders relevant hervorzuheben, während die Erkennung eines Eigenplagiates die Messung des Projekterfolgs nicht beeinflusst.

- Betrachtung aus einer rechtlich orientierten Sicht:
  - Es sind Plagiate mit möglichen rechtlichen und vertraglichen Konsequenzen zu erkennen.
  - Die Erkennung der Parodie wird wegen der Seltenheit in wissenschaftlichen Arbeiten ausgeschlossen.
  - Die Erkennung der Doppelschöpfung wird wegen des Bedarfs an den jeweils zusätzlichen juristischen Informationen ausgeschlossen.
  - Die Erkennung der unbewussten Entlehnung wird wegen des Bedarfs an den jeweils zusätzlichen juristischen Informationen ausgeschlossen.
  - Eigenplagiaten werden bei den aufbauenden Arten betrachtet.
- Betrachtung der grundlegenden Arten:
  1. Komplettplagiate müssen erkannt werden, da diese kritisch für den Projekterfolg sind.
  2. Verschleierungen müssen erkannt werden, da diese kritisch für den Projekterfolg sind.
- Betrachtung der von GuttenPlag gefundenen aufbauenden Arten:
  1. Halbsatzflickereien sollen erkannt werden, diese sind aber nicht kritisch für den Projekterfolg.
  2. Shake and Pastes sollen erkannt werden, diese sind aber nicht kritisch für den Projekterfolg.
  3. Alibi-Fußnoten sollen erkannt werden, diese sind aber nicht kritisch für den Projekterfolg.
  4. Strukturplagiate sollen erkannt werden, diese sind aber nicht kritisch für den Projekterfolg.
- Betrachtung der von GuttenPlag nicht gefundenen aufbauenden Arten:
  1. Die Erkennung von Eigenplagiaten hat keinen Einfluss auf den Projekterfolg.
  2. Die Erkennung von kopierten Zitaten hat keinen Einfluss auf den Projekterfolg.
- Betrachtung der Clone Types:
  1. Die Erkennung von Typ-1 Clones wird indirekt durch die Erkennung von Komplettplagiaten realisiert, aber nicht explizit als solches klassifiziert.

2. Die Erkennung von Typ-2 Clones wird indirekt durch die Erkennung von Verschleierungen realisiert, aber nicht explizit als solches klassifiziert.
3. Die Erkennung von Typ-3 Clones wird indirekt durch die Erkennung von Verschleierungen realisiert, aber nicht explizit als solches klassifiziert.
4. Die Erkennung von Typ-4 Clones wird indirekt durch die Erkennung von Verschleierungen realisiert, aber nicht explizit als solches klassifiziert.

## 1.6. Abgrenzung zu anderen Anwendungen

**Autor:** Marion Gottschalk
Eine umfassende Anforderungserhebung sieht vor, dass die Konkurrenzanwendungen von PlagTag betrachtet werden. Dies soll dazu dienen, weitere Ideen zu erhalten, welche in PlagTag umgesetzt oder nicht umgesetzt werden sollen. Somit lassen sich Anforderungen für die eigene Anwendung aus fremden Anwendungen ermitteln. Außerdem dient die Darstellung von Konkurrenzanwendungen der Abgrenzung zu diesen.

Aufgrund weniger Informationsquellen, die von Konkurrenzanwendungen bereitgestellt werden, muss die Betrachtung dieser Anwendung auf deren Visualisierung beschränkt werden. Allerdings lassen sich anhand der Visualisierungen für die Kunden leicht neue Anforderungen ermitteln, da sich der Funktionsumfang meist daraus ableiten lässt. Bei den Visualisierungen wird besonderer Wert auf die Ergebnisvisualisierung gelegt. Außerdem lassen sich Probleme in der Benutzerfreundlichkeit erkennen, die bei der Erstellung der eigenen Anwendung berücksichtigt werden.

Daher werden an dieser Stelle acht Konkurrenzanwendungen beschrieben. Um die Darstellung der Konkurrenzanwendungen vergleichbar zu halten, werden sie nach folgendem Muster beschrieben:

- **Übersicht:** Dieser Punkt beinhaltet allgemeine Angaben wie z.B. Titel, Anzahl der Worte im Text, Anzahl der Plagiate im Text, prozentuale Angaben zu den Plagiaten o.ä.
- **Überblick:** Im Vergleich zur Übersicht bezieht sich der Überblick auf die Visualisierung. Hier wird beschrieben, ob eine Visualisierung existiert, welche einen allgemeinen Überblick über das Ergebnis der Plagiatsüberprüfung gibt und wie diese realisiert ist.
- **Detail:** An dieser Stelle wird die Visualisierung der einzelnen Textpassagen, welche möglicherweise plagiiert sind, dargestellt.
- **Erklärung:** Unter diesem Punkt wird festgehalten, ob eine Beschriftung der angegebenen Werte vorhanden ist bzw. ein Benutzerhandbuch bereitgestellt wird.
- **Verständlichkeit:** Nachdem die Anwendungen getestet bzw. betrachtet wurden, soll hier mit einer Begründung beschrieben werden, ob die Ergebnisse der Anwendungen verständlich sind oder nicht.
- **Bewertung:** An dieser Stelle werden die Ergebnisse des *Plagiatserkennungstest 2010* von Katrin Köhler und Debora Weber-Wulff [KWW10] zusammengefasst, um eine Aussage zur Qualität der Anwendungen zu erhalten.

Die Referenzen in den Beschreibungen der Anwendungen beinhalten im Literaturverzeichnis einen Verweis zu den entsprechenden Demos, Screenshots oder Videos, um die Ergebnisse dieser Beschreibung nachzuvollziehen. Am Ende der Beschreibung von den Konkurrenzanwendungen werden die Ergebnisse daraus zusammengefasst und die entstandenen Anforderungen des Kundens an PlagTag aufgelistet.

### 1.6.1. Konkurrenzanwendungen

Acht Anwendungen zur Plagiatserkennung bzw. zur Clone-Erkennung werden nach dem oben beschriebenen Muster vorgestellt. Die sich daraus ergebenen *funktionalen Anforderungen* werden am Ende zusammengefasst und befinden sich außerdem im Kapitel 3.2.

#### PlagiatCheck

Bei *PlagiatCheck* [pla12b] handelt es sich um eine kostenlose Testversion von *PlagScan* [Pla12e]. Diese ermöglicht das Hochladen von Texten des Formates DOC/DOCX, HTML oder TXT bzw. die direkte Eingabe in ein Textfeld, um Texte für eine Plagiatsüberprüfung bereitzustellen.

- **Übersicht:** Es sind keine allgemeinen Informationen nach der Plagiatsüberprüfung vorhanden.
- **Überblick:** Es gibt keine visuelle Gesamtübersicht über gefundene Plagiate. Während der Plagiatsüberprüfung wird der Status der Überprüfung sowie die aktuell geprüfte Quelle angezeigt.
- **Detail:** Nach der Plagiatsüberprüfung wird eine Liste mit Quellen und die von dort plagiierte Textpassage ausgeben. Außerdem sind die Quellen mit einer *Relevanz* gekennzeichnet, dessen Bedeutung nicht ausreichend beschrieben ist.
- **Erklärung:** Bei dieser Anwendung ist nur wenig Dokumentation zu finden. Am Rand der Plagiatsüberprüfung wird ein Leitfaden und eine kurze Beschreibung zu den Ergebnissen geliefert.
- **Verständlichkeit:** Die Durchführung der Plagiatsüberprüfung ist einfach, allerdings sind die gelieferten Ergebnisse der Überprüfung nicht eindeutig zu interpretieren.
- **Bewertung:** In dem Plagiatserkennungstest wird diese Anwendung nicht aufgeführt. Allerdings wurde die kostenpflichtige Version *PlagScan* getestet und belegte Rang 4 von 26. Daher gilt *PlagScan* als teilweise nützlich laut Köhler und Weber-Wulff [KWW10].

#### PlagiarismFinder

Um sich einen Überblick über die Anwendung *PlagiarismFinder* machen zu können, wird eine Testversion zur Verfügung gestellt. Dabei handelt es sich um eine Desktop-Anwendung, in welche Texte des Formats PDF, TXT und DOC/DOCX geladen und überprüft werden können [pla12a].

- **Übersicht:** Die allgemeinen Informationen zur Visualisierung dieser Anwendung umfassen den Zeitpunkt der Überprüfung, Letzte Veränderung, Anzahl der Wörter und den Dateinamenpfad.
- **Überblick:** Zur Gesamtvisualisierung wird ein Überblick mit allen bisher geprüften Plagiatsdokumenten gegeben, wobei durch die Anzahl markierter Symbole angezeigt wird, wie wahrscheinlich es ist, dass es sich um ein Plagiat handelt.
- **Detail:** In der Detailansicht werden mögliche Plagiate farblich markiert. Dabei steht jede Farbe für jeweils eine Quelle. Zusätzlich wird der Text durch *Fettdruck* hervorgehoben, wenn die Wahrscheinlichkeit sehr hoch ist, dass es sich um ein Plagiat handelt.
- **Erklärung:** Die Ergebnisse der Plagiatsüberprüfung hingegen sind durch einen Anhang am Ergebnisdokument ausführlich erläutert.
- **Verständlichkeit:** Die Angaben in der Übersicht sind nicht selbsterklärend und es werden keine weiteren Informationen dazu geliefert. Die Plagiatsüberprüfung ist ansonsten einfach zu handhaben.

- **Bewertung:** Beim Plagiatserkennungstest erreichte diese Anwendung Rang 6 von 26.

**PlagAware**

*PlagAware* ist eine kostenpflichtige Anwendung, welche nur Screenshots ihrer Anwendung bereitstellt, um einen ersten Eindruck zu bekommen [Pla12c].

- **Übersicht:** Die allgemeinen Informationen zur Visualisierung dieser Anwendung umfassen den Zeitpunkt der Überprüfung, Anzahl der Zeichen und Wörter, Liste von Internetquellen.
- **Überblick:** Die erste Gesamtvisualisierung stellt den Plagiatskandidaten mit farblicher Hervorhebung der möglichen Plagiate dar. Außerdem werden die möglichen Quellen mit angegeben. Die zweite Gesamtvisualisierung des Plagiatskandidaten stellt sich in einem Strichcode dar, in welchem die verschieden plagiierten Textpassagen dargestellt sind. Dabei bezieht sich der Strichcode aber nur auf eine Quelle.
- **Detail:** In der Detailansicht werden pro Quelle die einzelnen Textpassagen im Plagiatskandidaten farblich markiert. Dieselbe farbliche Markierung wird in der Quelle verwendet, um die einzelnen Textpassen schnell miteinander vergleichen zu können.
- **Erklärung:** Die Informationen aus der Übersicht sind ausführlich beschrieben. Ein Handbuch zur Anwendung war nicht verfügbar.
- **Verständlichkeit:** Durch die Darstellung mit Hilfe des Strichcodes ist die Visualisierung leicht nachzuvollziehen.
- **Bewertung:** Beim Plagiatserkennungstest erreichte diese Anwendung Rang 1 von 26, da es bei der deutsch- als auch englischsprachigen Plagiatsüberprüfung zu den besten Anwendungen gehörte.

**Turnitin**

Bei *Turnitin* [tur12] handelt es sich um eine kostenpflichtige Anwendung, welche anhand von Videos vorgestellt wird.

- **Übersicht:** Die allgemeinen Informationen zur Visualisierung dieser Anwendung umfassen den Titel, Autoren und die Anzahl der Plagiate in Prozent.
- **Überblick:** Es existiert keine Gesamtvisualisierung außer dem Prozentwert für die Plagiate, welcher sich nach Veränderungen in der Detailansicht anpasst.
- **Detail:** In der Detailansicht werden die möglicherweise plagiierten Textpassagen farblich gekennzeichnet. Dabei wird für jede gefundene Quelle eine Farbe verwendet. Durch einen Mouseover über eine farblich markierte Textpassage öffnet sich ein Popup, in dem der Originaltext der Quelle dargestellt wird. So kann ein direkter Abgleich zwischen den beiden Textpassagen stattfinden und dann vom Anwender entschiedene werden, ob es wirklich ein Plagiat ist.
- **Erklärung:** Zur Veranschaulichung der Anwendung wird ein Video auf der Website zur Verfügung gestellt. Ein Benutzerhandbuch steht nicht zur Verfügung.
- **Verständlichkeit:** Im Video ausführlich beschrieben, daher scheint die Bedienung einfach zu sein. Keine genaue Aussage möglich.
- **Bewertung:** Beim Plagiatserkennungstest erreichte diese Anwendung Rang 2 von 26, da es bei der deutschsprachigen Plagiatsüberprüfung zu den besten Anwendungen gehörte.

### Urkund

Um sich einen Überblick über *Urkund* [urk12] zu verschaffen, wird eine Testversion mit Testdaten bereitgestellt. Unterstützt werden eine Vielzahl von Formaten wie z.B. DOC/DOCX, PDF und HTML [urk12].

- **Übersicht:** Die allgemeinen Informationen zur Visualisierung dieser Anwendung umfassen den Titel, den Prüfer und die Anzahl der Plagiate in Prozent.
- **Überblick:** Zur Gesamtvisualisierung der Ergebnisse wird eine Liste mit Internetquelle geliefert, welche von dem Anwender selbst an- und abgewählt werden können, wobei sich die Detailansicht anpasst.
- **Detail:** In der Detailansicht sind mögliche Plagiate am Rand der Textpassage farblich markiert. Dabei werden zwei Farben unterschieden, wobei zwischen Plagiat und *nicht Plagiat* unterschieden wird. Beim Auswählen einer Textpassage öffnet sich ein Popup, in dem der Originaltext abgebildet ist und dann entschieden werden kann, ob es ein Plagiat ist oder nicht. Es besteht auch die Möglichkeit, dass mehrere Quellen für eine Textpassage gefunden wurden, auch dies wird am Rand gekennzeichnet und der Anwender kann wählen, welche Quelle plagiiert wurde. Außerdem ist ein PDF-Export der Detailansicht möglich.
- **Erklärung:** Die Erklärungen sind sehr umfangreich in einem User Guide beschrieben.
- **Verständlichkeit:** Die Anwendung ist einfach zu verwenden, auch ohne den User Guide zu lesen.
- **Bewertung:** Beim Plagiatserkennungstest erreichte diese Anwendung Rang 5 von 26.

### Plagiarisma

*Plagiarisma* stellt eine Desktop-Anwendung bereit, in welche die zu überprüfenden Texte in ein Textfeld kopiert werden können [Pla12d].

- **Übersicht:** Die allgemeinen Informationen zur Visualisierung dieser Anwendung umfassen die Anzahl von Zeichen und Worte, die Anzahl der Plagiate in Prozent und die Auswahl wo gesucht werden soll. Dabei werden die Suchmöglichkeiten *Google* und *Scholar* unterschieden.
- **Überblick:** In der Gesamtübersicht werden alle selbst geschrieben Textpassagen des Plagiatskandidaten gekennzeichnet.
- **Detail:** In der Detailansicht werden unterschiedlich große Textpassagen untersucht und als *unique* bezeichnet oder eine Liste mit Links, welche womöglich plagiiert wurden, ausgegeben.
- **Erklärung:** Keine offensichtliche Erklärungen zur Anwendung vorhanden.
- **Verständlichkeit:** Bei der Durchführung der Suche ist es unklar wonach überhaupt gesucht wurde.
- **Bewertung:** Beim Plagiatserkennungstest erreichte diese Anwendung Rang 10 von 26.

### Ephorus

Bei *Ephorus* handelt es sich um eine kostenpflichtige Anwendung, welche zur Veranschaulichung nur Screenshots bereitstellt [eph12].

- **Übersicht:** Die allgemeinen Informationen zur Visualisierung dieser Anwendung umfassen den Titel, das Datum der Überprüfung und die Anzahl der Plagiate in Prozent.

- **Überblick:** Zur Gesamtvisualisierung der Ergebnisse wird eine Liste mit Internetquelle geliefert, welche von dem Anwender selbst an- und abgewählt werden können, wobei sich die Detailansicht anpasst.
- **Detail:** In der Detailansicht werden die zuvor ausgewählten Internetquelle mit dem Plagiatskandidaten verglichen und mögliche Plagiate mittels einer Farbe in beiden Texten markiert.
- **Erklärung:** Außer einer Kurzbeschreibung auf der Website keine weiteren Informationen zugänglich.
- **Verständlichkeit:** Der Einsatz ist trotz mangelnder Informationen einfach.
- **Bewertung:** Beim Plagiatserkennungstest erreichte diese Anwendung Rang 3 von 26.

#### ConQAT

*ConQAT* ist eine kostenlose Anwendung zur Clone-Erkennung [con12]. Daher wird nur die Visualisierung der Anwendung betrachtet.

- **Übersicht:** Die allgemeinen Informationen zur Clone-Erkennung umfassen hier den Titel, das Datum der Überprüfung und die Anzahl der Plagiate in Prozent.
- **Überblick:** Zur Gesamtvisualisierung wird eine Art Strichcode verwendet. Dieser unterteilt sich in die zwei Farben rot und grün, wobei rot für Plagiat und grün für kein Plagiat steht. Dies stellt das Verhältnis zu eigener Leistung und Plagiate dar.
- **Detail:** In der Detailansicht wird eine Matrix verwendet. Hierbei stellt jedes Rechteck in der Matrix eine Klasse dar und je nach Einfärbung ist zu erkennen, wie viel Plagiate ungefähr in der Klasse vorhanden sind. Durch einen Mouseover werden dann genaue Informationen zu der Plagiatsanzahl gegeben.
- **Erklärung:** Auf der Website von ConQAT wird ein Benutzerhandbuch zur Verfügung gestellt.
- **Verständlichkeit:** Die Visualisierung der Anwendung ist auch ohne Dokumentation leicht verständlich.
- **Bewertung:** Nicht vorhanden.

### 1.6.2. Ergebnisse der Protokolle

An dieser Stelle sollen die Ergebnisse der Gesamt- und Detailansicht zusammengefasst werden.

#### Gesamtansicht

In der Gesamtansicht, welche sich nach der Plagiatsüberprüfung ergibt, werden von den Konkurrenzanwendungen drei wesentliche Visualisierungen verwendet:

- **Strichcode:** Mit Hilfe eines Strichcodes werden die Menge und die Position der Plagiate im Text deutlich. Dabei werden verschiedene Farben verwendet, wobei jede Farbe für eine andere plagiierte Textpassage steht, somit wird auch die Länge der Plagiate anhand des Strichcodes deutlich. Diese Darstellungsform wird z.B. bei PlagAware [Pla12c] verwendet und kann in Abbildung 1 betrachtet werden.
- **Liste der gefunden Quellen:** Neben der Detailansicht ist eine Liste mit allen gefunden Quellen und den daraus plagiierten Textpassagen eine gute Ergänzung. In dieser Liste sollte die Möglichkeit bestehen, die einzelnen Quellen an- und abzuwählen, falls es sich bei dem gefundenen Plagiat doch nicht um ein Plagiat handelt. Anhand dieser Liste können alle möglichen Plagiate des Textes schnell durchgegangen und überprüft werden.

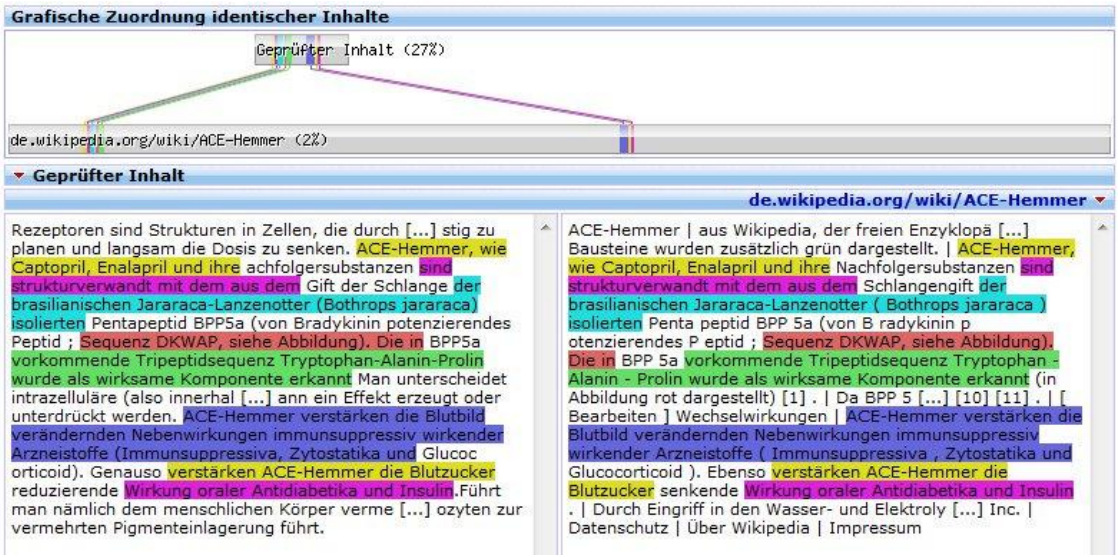
Grafische Zuordnung identischer Inhalte
Geprüfter Inhalt (27%)
de.wikipedia.org/wiki/ACE-Hemmer (2%)
Geprüfter Inhalt
de.wikipedia.org/wiki/ACE-Hemmer
Rezeptoren sind Strukturen in Zellen, die durch [...] stig zu planen und langsam die Dosis zu senken. ACE-Hemmer, wie Captopril, Enalapril und ihre achfolgersubstanzen sind strukturverwandt mit dem aus dem Gift der Schlange der brasilianischen Jararaca-Lanzenotter (Bothrops jararaca) isolierten Pentapeptid BPP5a (von Bradykinin potenzierendes Peptid ; Sequenz DKWAP, siehe Abbildung). Die in BPP5a vorkommende Tripeptidsequenz Tryptophan-Alanin-Prolin wurde als wirksame Komponente erkannt Man unterscheidet intrazelluläre (also innerhal [...] ann ein Effekt erzeugt oder unterdrückt werden. ACE-Hemmer verstärken die Blutbild verändernden Nebenwirkungen immunsuppressiv wirkender Arzneistoffe (Immunsuppressiva, Zytostatika und Glucoc orticoid). Genauso verstärken ACE-Hemmer die Blutzucker reduzierende Wirkung oraler Antidiabetika und Insulin.Führt man nämlich dem menschlichen Körper verme [...] ozyten zur vermehrten Pigmenteinlagerung führt.
ACE-Hemmer | aus Wikipedia, der freien Enzyklopä [...] Bausteine wurden zusätzlich grün dargestellt. | ACE-Hemmer, wie Captopril, Enalapril und ihre Nachfolgersubstanzen sind strukturverwandt mit dem aus dem Schlangengift der brasilianischen Jararaca-Lanzenotter ( Bothrops jararaca ) isolierten Penta peptid BPP 5a (von B radykinin p otenzierendes P eptid ; Sequenz DKWAP, siehe Abbildung). Die in BPP 5a vorkommende Tripeptidsequenz Tryptophan - Alanin - Prolin wurde als wirksame Komponente erkannt (in Abbildung rot dargestellt) [1] . | Da BPP 5 [...] [10] [11] . | [ Bearbeiten ] Wechselwirkungen | ACE-Hemmer verstärken die Blutbild verändernden Nebenwirkungen immunsuppressiv wirkender Arzneistoffe ( Immunsuppressiva , Zytostatika und Glucocorticoid ). Ebenso verstärken ACE-Hemmer die Blutzucker senkende Wirkung oraler Antidiabetika und Insulin . | Durch Eingriff in den Wasser- und Elektroly [...] Inc. | Datenschutz | Über Wikipedia | Impressum

**Abbildung 1:** Strichcode von Plagaware [Pla12c]

- **Quellenübersicht:** Diese Visualisierung beschreibt die Darstellung des gesamten Textes, in dem alle möglichen Plagiate farblich gekennzeichnet sind. Neben der farblichen Darstellung werden die jeweiligen Quellen zu einzelnen Textpassagen angegeben. Dadurch lässt sich auf einem Blick erkennen, wie viel womöglich plagiiert wurde und woher es stammt.

#### Detailansicht

In der Detailansicht sollen die einzelnen plagiierten Textpassagen hervorgehoben werden, dabei werden ebenfalls drei wesentliche Visualisierung verwendet:

- **Farbliche Hervorhebung:** Die erste Visualisierung kennzeichnet den womöglich plagiierten Text mit einer Farbe. Welche Quelle plagiiert wurde, wird in einer Tabelle dargestellt, indem jeder Quelle eine Farbe zugeteilt ist. Dieselbe Farbe wird dann für plagiierte Textpassagen aus dieser Quelle verwendet.
- **Gegenüberstellung:** Bei der Gegenüberstellungen wird je eine Quelle mit dem geprüften Text verglichen. Dazu werden plagiierte Textpassagen in einer Farbe markiert und dieselbe Farbe bei den Textpassagen der Quelle angewandt, um die Gemeinsamkeiten schnell herausstellen zu können.
- **Popup:** Eine weitere Form der Visualisierung ist der Einsatz von Popups. Im Text werden alle Textpassagen, die womöglich plagiiert sind, markiert. Werden diese Textpassagen ausgewählt, öffnet sich ein Popup mit dem Originaltext der Quelle. Somit können alle plagiierten Textpassagen gekennzeichnet werden, auch wenn unterschiedliche Quellen zugrunde liegen.

### 1.6.3. Anforderungen an PlagTag

Abschließend haben sich aus der Vorstellung der konkurrierenden Anwendungen und der Diskussion mit den Kunden Andreas Winter und Jan Jelschen folgende Anforderungen für die Anwendung PlagTag ergeben. Diese Anforderungen werden in Abschnitt 3.7 erneut aufgegriffen.

- Der Status der Verarbeitung von verschiedenen Überprüfungen kann in Form einer Auflistung eingesehen werden, wie z.B. in Abbildung 2.
- Das Webfrontend muss ein Benutzerhandbuch für die Verwendung von PlagTag bereitstellen, da erkannt wurde, dass sich die Verwendung der Konkurrenzanwendungen schwer gestaltete aufgrund fehlender Informationen, wie z.B. bei Plagiarisma [Pla12d].

01 Ahornsirup.doc 1 542 2009/02/12 Report fertig XX245B1202091
01 Ahornsirup.docx 1 542 2009/01/14 Report fertig XX116B1401091
02 Was ist Ajax.docx 1 753 2009/01/14 Report fertig XX126B1401091
03 Avignon.docx 1 700 2009/01/14 Report fertig XX122B1401091
04 Bedürfnisanstalten.docx 1 614 2009/01/14 Report fertig XX124B1401091

**Abbildung 2:** Status der Überprüfung [Pla12d]

- PlagTag muss dem Anwender eine Auswahl der Sprache des Plagiatskandidaten ermöglichen, da dies die meisten Konkurrenzanwendungen anbieten. Dabei wird sich aber auf die Sprachen deutsch und englisch beschränkt.
- PlagTag muss eine Gesamtübersicht in Form eines Strichcodes ermöglichen, wie es bei GuttenPlag [Div11] möglich ist (siehe Abbildung 3).

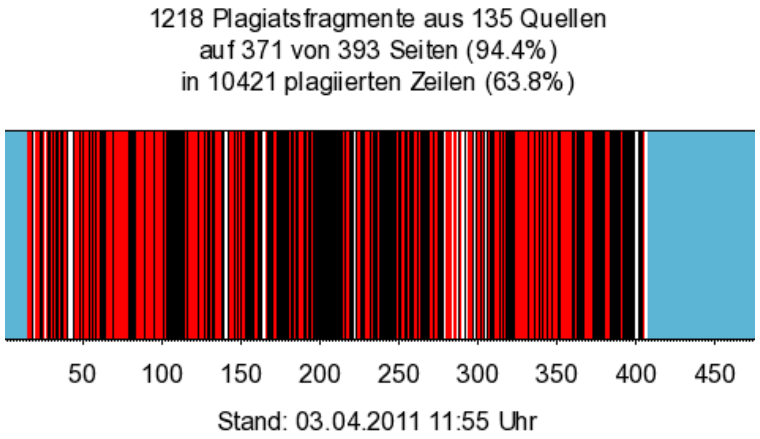
1218 Plagiatsfragmente aus 135 Quellen
auf 371 von 393 Seiten (94.4%)
in 10421 plagiierten Zeilen (63.8%)
50 100 150 200 250 300 350 400 450
Stand: 03.04.2011 11:55 Uhr

**Abbildung 3:** Strichcode auf GuttenPlag [Div11]

- PlagTag muss eine Gesamtübersicht über die gefundenen Plagiate in Form eines Kuchendiagramms bieten, wobei das Verhältnis zwischen Plagiaten und eigener Leistung dargestellt wird.
- Das Ergebnisdokument muss den Originaltext aus den Referenzdaten dem möglichen Plagiat gegenüberstellen, wie z.B. bei Urkund [urk12] oder Turnitin [tur12].
- Gefundene Plagiate müssen vom Anwender bei der Visualisierung zu deaktivieren sein, wie z.B. bei Urkund [urk12].
- Die Referenzdaten sollen bei der Visualisierung abwählbar sein, d.h. das einzelne Referenzdaten mit dem Plagiatskandidat gegenübergestellt werden können, wie z.B. bei Urkund [urk12].
- PlagTag muss eine eigene Abschätzung abliefern, ob es sich um ein Plagiat handelt, wie z.B. bei Plagiarisma (siehe Abbildung 2).
- Das Ergebnisdokument muss die möglichen Plagiate des Plagiatskandidat farblich hervorheben, siehe Abbildung 1, wobei zwei Farben ausreichen sollen, um Komplettplagiate und Verschleierung unterscheiden zu können.
- Das Ergebnisdokument sollte als PDF exportierbar sein, um dieses abspeichern zu können.

### 1.6.4. Inhalte des Ergebnisdokuments

Anhand der Anforderungserstellung für PlagTag in Abschnitt 1.6.3, hat sich eine genaue Definition der Visualisierung des Ergebnisdokuments ergeben. Das Dokument ist in die Bereiche *grafische Visualisierung*, *Informationen der Plagiatsüberprüfung* und *Ergebnisse der Plagiatsüberprüfung*. Der Anwender hat die Möglichkeit sich seine grafische Visualisierung auswählen zu können, diese steht am Dokumentanfang. Die Grafik umfasst die Anzahl der gefundenen Plagiate. Die Informationen der Plagiatsüberprüfung umfassen den Namen der Plagiatsüberprüfung, den Namen des Überprüfers sowie das Start- und Enddatum. Des Weiteren werden die hochgeladenen Referenzdokumente und die selektierten Internetquellen aufgelistet. Darüber hinaus wird eine Liste der ausgewählten Algorithmen und weiteren Einstellungen angezeigt. Der dritte Bereich, die Ergebnisse der Plagiatsüberprüfung, enthält die plagiierten Textpassagen, welche als Komplettplagiate rot markiert sind. Die gefundenen Plagiate sind mit einer Fußnote versehen in welcher die Referenz auf das plagiierte Dokument steht. Zur Übersichtlichkeit soll das Ergebnisdokument Seitenzahlen und einen Verweis auf PlagTag enthalten.

## 1.7. Vorgehensmodell

**Autor:** Marion Gottschalk, Sieglinde Hahn

Das Vorgehen innerhalb der PG soll während der Entwicklung agil und iterativ sein, wie es in Abbildung 4 dargestellt ist. Zu Beginn der PG richtet sich das Vorgehen nach dem *Wasserfallmodell* [Bal09], da zuerst ein initiales Pflichtenheft und ein erster Entwurf der Anwendung erstellt werden soll. Erst nach diesen zwei Phasen beginnt das agile und iterative Vorgehen, wobei das Pflichtenheft und der Entwurf rückwirkend angepasst werden können. Es werden insgesamt drei Zyklen, im folgenden als Versionen bezeichnet, des Entwurfs durchlaufen, wobei die Phasen *Analyse und Design*, *Implementierung*, *Test und Reflexion* und *Fertigstellung* immer wieder auftreten werden. Die Phasen *Analyse und Design*, *Implementierung* und *Test und Reflexion* finden in jedem Zyklus parallel statt.

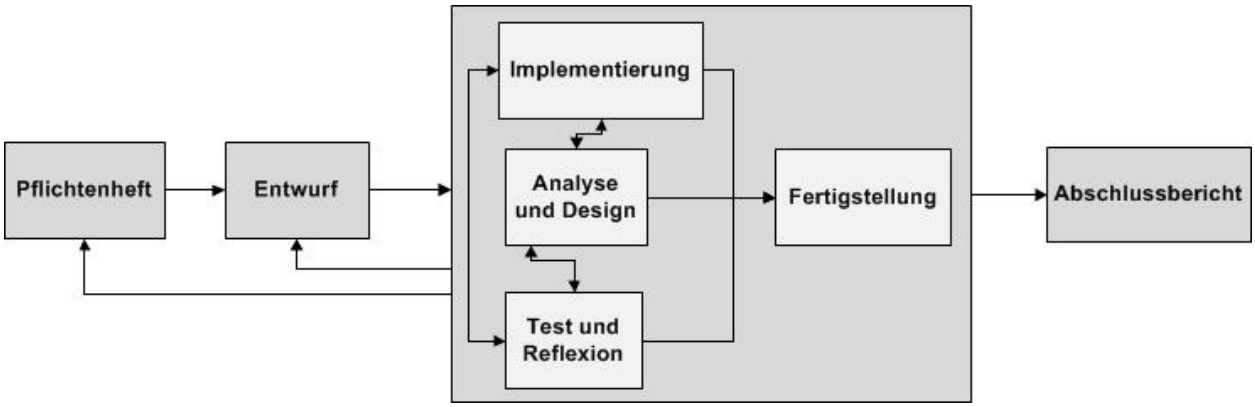


**Abbildung 4:** Vorgehensmodell

Die einzelnen Phasen werden wie folgt definiert:

- **Analyse und Design:** In dieser Phase werden die Anforderungen bestimmt, welche in diesem Zyklus realisiert werden sollen. Dabei werden die Anforderungen mit der höchsten Priorität, also den Pflichtanforderungen, aus dem Pflichtenheft entnommen. Diese Anforderungen werden im weiteren Vorgehen in Komponenten eingeteilt oder bestehenden Komponenten zugeordnet. Dabei sollen Alternativen berücksichtigt und evaluiert werden. Die erstellte Struktur aus dem ersten Zyklus wird in den weiteren Zyklen erweitert, um den Bezug zur bisherigen Architektur beizubehalten.

- **Implementierung:** In dieser Phase werden die Komponenten, welche parallel in der Phase *Analyse und Design* bestimmt werden, realisiert und mit den bestehenden Komponenten verbunden. Es findet ein ständiger Austausch mit der Phase *Test und Reflexion* und *Analyse und Design* statt.
- **Test und Reflexion:** Die Implementierung der Komponenten werden in dieser Phase getestet, um die Korrektheit der Komponenten zu gewährleisten. Außerdem wird überprüft, ob alle Anforderungen, welche in der Phase *Analyse und Design* bestimmt wurden, entsprechend der Beschreibung im Pflichtenheft realisiert wurden. Gefundene Probleme werden innerhalb der Phase *Implementierung* behoben.
- **Fertigstellung:** In dieser Phase werden die erstellte Funktionalität dem Kunden vorgestellt und übergeben.

Über die Änderungen von Versionen hinweg und innerhalb der einzelnen Phasen wird das Pflichtenheft parallel zu getätigten Änderungen angepasst.

Die Arbeitsweise in der PG soll in vielen Teilbereichen so sein, dass wenig bürokratischer Aufwand und Regeln benötigt werden. Dabei wird auf folgende Gesichtspunkte Wert gelegt:

- effiziente Kommunikation durch ein Ticketsystem und Projektleitung,
- Dokumentation so viel wie nötig, so wenig wie möglich (KISS - keep it simple, stupid),
- enge Zusammenarbeit mit dem Auftraggeber (z.B. Anwesenheit bei der Mehrzahl der Besprechungen),
- relativ kurze Zeiten zwischen Meilensteinen (meist ca. 6 Wochen) und kundenbezogene Entwicklung, gemäß Anforderungen der Kunden.

### 1.7.1. Projektmanagement

Damit die Arbeit während des Projekts leichter zu koordinieren ist, wird das Projektmanagement-Werkzeug Trac benutzt. Dieses Werkzeug bietet die Möglichkeit, Aufgaben mithilfe eines Ticketsystems zu vergeben. Ein Ticket beinhaltet eine Aufgabendefinition sowie Daten zu der Aufgabe. Diese Tickets können dann Projektteilnehmern zugewiesen werden. Außerdem stellt trac ein Wiki zur Verfügung, welches zum Erstellen und Verwalten von Protokollen und anderen Dokumenten genutzt wird. Auch das Projekthandbuch wird im Wiki angelegt, bis dieses zu einem späteren Zeitpunkt in die Übergabe, in Form der Abschlussdokumentation, einfließt.

Im Projekthandbuch werden verschiedene Sachverhalte festgelegt, die mit dem Ablauf der PG in Verbindung stehen. Beispielsweise sind Rollen, die einzelne Teilnehmer einnehmen und die damit verbundenen Verantwortungen festgehalten, aber auch Regeln für die Zusammenarbeit in der Gruppe, sowie deren Sanktionen bei Nichteinhaltung, werden hier festgelegt.

Des Weiteren wird als Software zur Versionsverwaltung Apache *Subversion* (Subversion-Repository) eingesetzt. Dies ermöglicht es, allen Teilnehmern am selben Projekt zu arbeiten und sichert die verschiedenen Versionen des Projekts auf einem Server der Universität Oldenburg. Im Subversion-Repository befindet sich unter anderem auch ein Ordner zur Abgabe von Dokumenten und Programmen an den Kunden, in dem aber auch Zwischenabgaben zur Verfügung gestellt werden.

### 1.7.2. Meilensteine

Die Umsetzung und Erreichung der Meilensteine ist im Fortschrittsbericht dokumentiert. Die Zeitplanung sieht folgende Meilensteine vor:

- Tag der Informatik: 05.03.2012
  Zur Vorstellung der PG wird am Tag der Informatik ein Infostand vorbereitet, an diesem soll der aktuelle Stand der PG vorgestellt werden. In welcher Form diese Vorstellung stattfindet, wird kurzfristig entschieden.

- Version 0: 16.04.2012
  Im Rahmen der Machbarkeitsstudie wird ein technischer Durchstich erstellt. Dabei sollen verschiedene Komponenten miteinander kommunizieren können und die Datenhaltung in einer einfachen Form realisiert sein. Außerdem soll eine Anfrage des Webfrontends an die Datenhaltung ermöglicht werden (z.B. Abfrage eines Benutzers beim Einloggen).

- Version 1: 24.05.2012
  In *Version 1* muss ein vertikaler Durchstich durch die Architektur implementiert werden, das heißt jede Komponente von PlagTag wird in einer minimalen Form vertreten sein. Hierbei soll ein einfach umzusetzender Algorithmus zur Plagiatsüberprüfung zur Erkennung des Plagiatstypen Komplettplagiat implementiert werden. Eine erste Visualisierung zur Plagiatsüberprüfung im Rahmen einer Textgegenüberstellung wird realisiert. Die Plagiatskandidaten und Referenzdokumente werden als PDF und TXT bereitgestellt und verglichen. Die Datenhaltung soll das Speichern von Plagiatsüberprüfungen ermöglichen. Außerdem wird die Internetrecherche ohne Selektion von Internetquelle und mit einem einfachen Algorithmus zur Bestimmung von Suchbegriffen implementiert. Die Ausgabe erfolgt in einem Web-Frontend.

- Version 2: 09.07.2012
  In *Version 2* wird es möglich sein sich gefunden Plagiate visuell als Kuchendiagramm und Strichcode ausgeben zu lassen. Es wird mindestens ein weiterer Algorithmus zur Erkennung von Komplettplagiaten implementiert. Zusätzlich soll mindestens ein linguistisches Verfahren umgesetzt werden, welches generisch sein muss. Es muss eine automatische Internetquellensuche möglich sein. Die Internetquelle müssen selektierbar sein. Des Weiteren muss die Internetrecherche abgebrochen werden können. Das Ergebnisdokument muss Informationen über die Position der Plagiatskandidaten enthalten. Es soll dem Anwender möglich sein zwischen dem Experten- und Standardmodus zu wechseln. Im Expertenmodus soll die Auswahl von Algorithmen möglich sein. Dem Anwender soll die Möglichkeit geboten werden manuell die Sprache festzulegen, zu diesem Zweck soll englisch unterstützt werden. Der Plagiatstyp Verschleierung muss erkannt werden können und farblich von einem Komplettplagiat unterschieden werden können. Es muss die Statushistorie der Plagiatsüberprüfung anzeigen können. Nach der Plagiatsüberprüfung soll eine Abschätzung geliefert werden können, ob es sich bei einem Plagiatskandidat um ein Plagiat handelt. Darüber hinaus soll das Ergebnisdokument als PDF exportierbar sein. Das Verfahren zur Erkennung von Plagiaten kann abgebrochen werden. PlagTag muss die Möglichkeit bieten aus dem Webfrontend Plagiatsüberprüfungen zu löschen. Personenbezogene Daten zum Autor des Plagiatskandidaten müssen anonymisiert werden. Die Referenzdaten müssen nach Beendigung der Überprüfung gelöscht werden. Abschließend müssen Plagiatskandidaten nach der Überprüfung gelöscht werden.

- Version 3: 31.08.2012
  In *Version 3* werden weitere Algorithmen zur Erkennung der Plagiatstypen Halbsatzflickereien, Shake and Pastes, Alibi-Fußnoten, Strukturplagiate und Eigenplagiate implementiert. Des Weiteren sollen kopierte Zitate erkannt werden können. PlagTag muss die verwendete Sprache automatisch erkennen. Die in der Informatik verwendet Zitierweise und die Zitierweise wie sie nicht in der Informatik verwendet werden müssen erkannt werden. PlagTag muss automatisch eine Widerlegung des Plagiatsverdachts durch Auswertung der Referenzdaten ermöglichen. Das Literaturverzeichnis muss bei der Referenzsuche berücksichtigt werden. Der Anwender soll die Möglichkeit haben gefundene Plagiate zu selektieren und zu entscheiden, ob es sich um Plagiate handelt. Rechenintensive Operationen müssen auf dem hochschuleigenen Clus-

tersystem verteilt ausgeführt werden. Die Kommunikation von PlagTag mit dem Cluster soll über SSH erfolgen. Der Zugriff auf den Cluster muss aus dem Uni-Netzwerk möglich sein.

- Abschlussbericht: 31.08.2012
  Die Phase *Abschlussbericht* verläuft parallel zu *Version 3*. In dieser Phase sollen die erstellten Dokumente *Pflichtenheft*, *Entwurfsdokument* und *Testdokument* überarbeitet und zusammengefasst werden. Darüber hinaus werden Dokumente, welche für die Software-Übergabe benötigt werden erstellt. Diese Dokumente wurden mithilfe des Deployment-Vortrags bestimmt, siehe 1.8. Zusätzlich werden Tests der fertigen Anwendung durchgeführt. Auch eine Überprüfungen der Qualität der Analysen während des Entwicklungsprozesses wird in dem abschließenden Bericht zur PG aufgenommen.

- Endpräsentation: 05.10.2012
  Die Endpräsentation beinhaltet die Übergabe von PlagTag an den Kunden. Es findet eine Präsentation vor den Stakeholdern statt. Zusätzlich werden weitere Personen eingeladen, welche im direkten oder indirekten Zusammenhang mit der PG stehen.

## 1.8. Deployment

**Autor:** Sieglinde Hahn
*Deployment* umfasst alle Aktivitäten zur Übergabe der Software [IA]. Die Übergabe von PlagTag stellt den Abschluss der PG dar, allerdings nicht den Abschluss von PlagTag. Der folgende Abschnitt befasst sich mit der zu erstellenden Dokumentation für die Übergabe von PlagTag. Gemäß unserer Anforderung *E140* muss die Erweiterbarkeit und Austauschfähigkeit einzelner Komponenten bestehen. Um diese Erweiterbarkeit und Austauschfähigkeit gewährleisten zu können, muss PlagTag ausreichend dokumentiert werden, um von späteren Entwicklern genutzt werden zu können.

In der Praxis existieren verschiedene Beispiele, welche Dokumente bei der Übergabe zu erstellen sind, daher werden zu Beginn dieses Abschnitts Praxisbeispiele von IBM, Harry Sneed, Jürgen Scheibl und der Deutschen Industrienorm angeschaut. Basierend auf diesen vorgestellten Praxisbeispielen, wird am Ende ein Fazit gezogen welche Dokumente bei der Übergabe von PlagTag zur Verfügung stehen müssen.

### 1.8.1. IBM

IBM [IBM02] nutzt für ihr Deployment das sogenannte Application Maintenance Turnover Package (AMTP), dieses enthält eine Checkliste für die Anwendungspflege und Erweiterung. Hierbei findet die Unterteilung in die Funktionen *Wartung und Erweiterung*, *Betrieb* und *Schulung und Training* statt. AMTP konzentriert sich nur auf die Anwendungspflege und Erweiterung. Ziel ist eine gute Kommunikation zwischen dem Entwicklungs- und Wartungsteam. Um eine solche Kommunikation zu erreichen werden folgende Bestandteile bei der Übergabe benötigt. Die Systembeschreibung und der gesamte Quellcode inklusive Kommentare ist auszuliefern. Die Impact Analyse wird benötigt, um die einzelnen Anforderungen und deren Auswirkungen analysieren zu können. Eine Reflektion der identifizierten Auswirkungen der Impact Analyse wird in der Komponentenbeschreibung festgehalten. Der Leistungsumfang der Software ist in einem Projekthandbuch zu dokumentieren. Dokumentierte Testverfahren wie durchgeführte Regressions- und Systemtest sind ebenfalls zu beschreiben. Abschließend ist eine Liste von aufgetretenen Fehlermeldungen zu erfassen, in welcher die Fehlerbedingung und Lösungsansätzen festgehalten sein sollen.

Die erstellte Abnahmecheckliste unterteilt sich in die Bereiche *Dokumentation*, *Anwendungsumgebung*, *Anwendung-Wartungstools* und *Status der Anwendung*. Die Checkliste überprüft im Bereich *Dokumentation*, ob die Anwendungsdokumentation vollständig und Verweise auf weiter Dokumente

vorhanden sind. Die Vollständigkeit beinhaltet auch die Korrektheit, Konsistenz und Nachvollziehbarkeit der Dokumente. Die *Anwendungsumgebung* umfasst die Überprüfung der Architektur und die Wartungsmöglichkeiten. Darüber hinaus wird überprüft, ob spezielles technisches Wissen erforderlich ist, um die Software zu Bedienen. Kundenspezifische Tools für die Wartung werden im Bereich *Anwendung-Wartungstools* geprüft. Die notwendige Konfiguration der Anwendung ist zu überprüfen, welche optimaler Weise in Form eines Konfigurationsdokument festgehalten werden sollte. Des Weiteren wird überprüft, ob die Anwendung ausreichend für die Inbetriebnahme im Bereich Pflege und Erweiterung ist. Abschließend wir überprüft, ob im Bereich *Status der Anwendung* bekannte Fehler und Schwachstellen dokumentiert sind.

### 1.8.2. Rational Unified Process

Der Rational Unified Process (RUP) ist ein Software-Engineering-Prozess, welcher den gesamten Lebenszyklus der Softwareentwicklung abdeckt. RUP vereint gesammelte Best Practices, hierbei wird ein Projekt in vier Phasen unterteilt, siehe Abbildung 5, welche von Philippe Kruchten dokumentiert wurden [Kru98]. Diese Phasen umfassen die *Konzeptionsphase*, *Entwurfsphase*, *Konstruktionsphase*

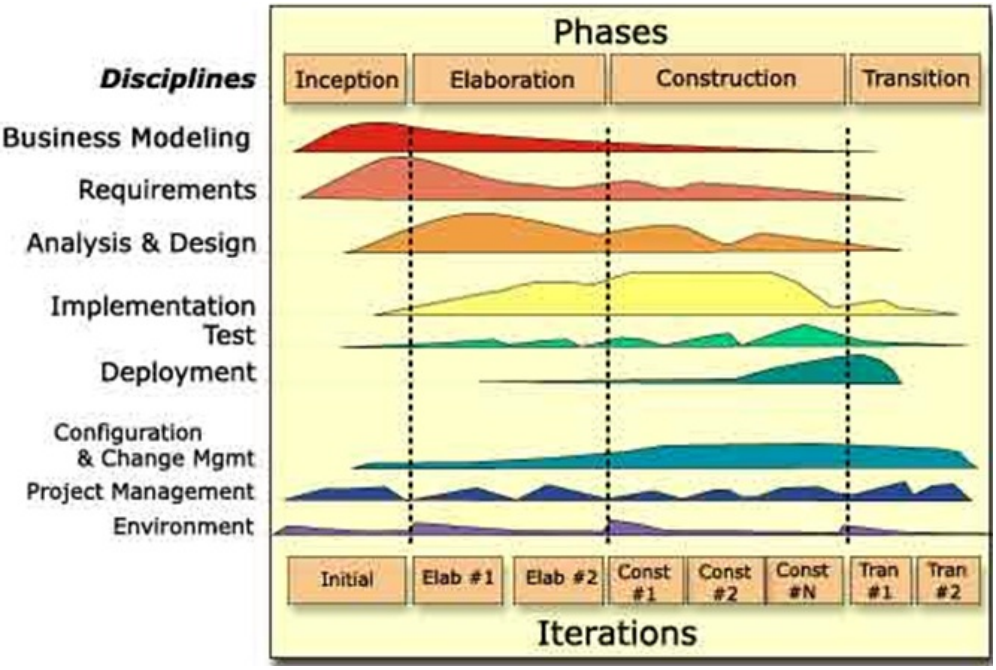


**Abbildung 5:** RUP Phasen [Tho08]

und *Übergabephase*. Da in diesem Abschnitt das Thema Deployment behandelt wird, betrachten wir nur den Bereich Deployment von RUP. Anhand der Abbildung 5 lässt sich erkennen, dass Deployment in den zu Beginn in der *Entwurfsphase* betrachtet wird. Eine zunehmende Betrachtung des Deployment findet in der *Konstruktionsphase* statt bis hin zur *Übergabephase*.

Der Deployment-Workflow von RUP umfasst nach Kruchten [Kru98] das Testen, die Paketierung, die Verteilung und die Installation der Software. Darüber hinaus muss es in diesem Workflow Schulungen für den Endanwender geben und eine Migration in bestehende Systeme bedacht werden. Die jeweiligen Aktivitäten sind abhängig von der Projektgröße. Die Hauptbestandteile bei der Auslieferungen ist somit eine lauffähige Software mit allen notwendigen Bestandteilen. Darüber hinaus werden Installationshinweise, Release- und Support-Hinweise und Schulungsunterlagen benötigt.

Dokumente sind bei RUP einzelnen Stakeholdern zuzuordnen wie in Abbildung 6 zu entnehmen. Dem Technischen Autor wird hier das Support Material zugeordnet, welches im bei der Einarbeitung in die Software behilflich sein soll. Das Support Material umfasst die Handbücher für die

Endanwender. Der sogenannte Verteilungsmanager ist für den Übergabeplan verantwortlich, dieser beschreibt wie das Produkt zum Kunden geliefert werden soll. Der Entwickler ist für die Erstellung der Schulungsunterlagen verantwortlich.

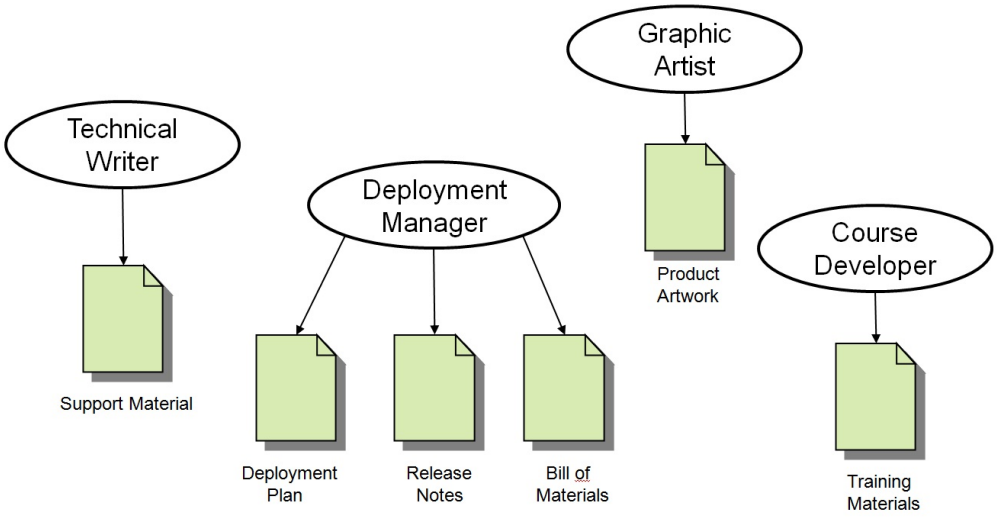


**Abbildung 6:** RUP Stakeholder angelehnt an [Kru98]

### 1.8.3. Harry Sneed

Das Fachgebiet von Harry Sneed umfasst u.a. das Software-Engineering, daher findet eine genauere Betrachtung seiner Vorgehensweise bei der Auslieferung einer Software statt. Basierend auf dem Buch [Sne04] von Harry Sneed, wurde betrachtet, welche Dokumente bei der Auslieferung einer Software relevant sind. Unterschieden wird hierbei Dokumentation, welche vor der Lieferung und nach der Lieferung erbracht werden muss. Die *Dokumentation vor der Lieferung* dient dazu, seine Systeme rechtzeitig umzustellen und mögliche Änderungen der Software anpassen zu können. *Dokumentation bei der Lieferung* dient dazu das System wie es in seiner Betriebsumgebung funktionieren soll, zu beschreiben. Des Weiteren dient diese Form der Dokumentation dazu Mängel der letzten Testzyklen in dokumentierter Form zu übergeben, um mögliche Fehler schnell beheben zu können. Zusätzlich dazu ist eine Liste über alle gemachten Änderungen über die Versionen hinweg zu dokumentieren.

Laut Sneed wird bei der Auslieferung der Software und der Dokumente unterschieden, ob es sich um eine inkrementelle Auslieferung oder eine Vollauslieferung handelt. Dabei findet eine Unterscheidung in die drei Bereiche der Dokumentation statt. Dokumentation kann technisch, fachlich oder nutzungsbezogen sein.

### 1.8.4. Hans-Jürgen Scheibl

Prof. Dr.-Ing. Hans-Jürgen Scheibl ist Professor an der Hochschule für Technik und Wirtschaft Berlin. Sein Fachgebiet umfasst u.a. das Software-Engineering. 1985 hat Herr Scheibl ein Buch zur Dokumentation von Softwareprojekten verfasst, auf diesem basieren die folgenden Inhalte [Sch85].

Die Gesamtdokumentation lässt in zwei Bereiche untergliedern, in *Technische Dokumentation* und *Projektbegleitende Dokumentation*. Der *Projektbegleitenden Dokumentation* kommt eine Sonderstellung zu, da diese zwar für den Ablauf der Projektabwicklung relevant ist, nach Abschluss des Projekts aber nur eine geringe Bedeutung gegenüber der *Technischen Dokumentation* besitzt. Die *Technische Dokumentation* lässt sich untergliedern in *Entwicklungsdokumentation*, *Produktdokumentation* und *Benutzerdokumentation*. Diese Unterteilung ist in der Abbildung 7 zu sehen.

Die *Entwicklungsdokumentation* umfasst hierbei Dokumente, welche zur Beschreibung der eigentlich Aufgabenstellung und deren erste Lösungsansätze dienen. Bei der *Produktdokumentation* wird das eigentliche Produkt dokumentiert, welche u.a. die Hardwarebeschreibung beinhaltet. Der letzte Bereich der *Technischen Dokumentation* ist die *Benutzerdokumentation*, welche die Inbetriebnahme des Produkts dokumentiert sowie Wartungs- und Fortbildungsunterlagen.

| **Projektdokumentation** | **Technische Dokumentation** | | |
|---|---|---|---|
| | **Entwicklungsdokumentation** | **Produktdokumentation** | **Benutzerdokumentation** |
| *Dokumente zur Projektentwicklung* | *Aufgabenbeschreibung* | *Übersichtsbeschreibung* | *Betriebsunterlagen* |
| Projektbeschreibung | Aufgabenstellung | Systembeschreibung | Benutzerhandbuch |
| Projektorganisation | Anforderungen | Funktionsweise | Bedienungsanleitung |
| Verantwortlichkeiten | Situationsanalyse | *Programmbeschreibung* | Operatorenhandbuch |
| Einsatzplanung | Problemanalyse | Programmumgebung | *Wartungsunterlagen* |
| Zeitplanung | Prozessbeschreibung | Programmstruktur | Installationsanleitung |
| Kostenplanung | Studien | Programmablauf | Fehlerbeschreibung |
| Dokumentationsplanung | *Lösungsbeschreibung* | Datenmodell | *Schulungsunterlagen* |
| Hilfsmittelaufstellung | Entwurfsschritte | Leistungskenndaten | Einführungen |
| Entwicklungsrichtlinien | Entwurfsalternativen | *Hardwarebeschreibung* | Ausbildungspläne |
| *Dokumente zur Entschei-* | Entwurfsentscheidungen | Anlagenübersicht | Trainingsunterlagen |

**Abbildung 7:** Dokumentationsverfahren nach Scheibl angelehnt an [Sch85]

Scheibl definiert für unterschiedliche Dokumente verschiedene Stakeholder, siehe Abbildung 8. Die Dokumentation ist in die bereits vorgestellten Bereiche der *Projektbegleitende Dokumentation* und *Technisches Dokumentation* untergliedert. Hier wurde speziell einzelnen Personengruppen zugeordnet für welche Dokumente diese laut Scheibl verantwortlich sind.

| | Produktdokumentation | Entwicklungsdokumentation | Projektdokumentation | Benutzerdokumentation |
|---|---|---|---|---|
| Anwendungsprogrammierer | X | X | X | X |
| Arbeitsvorbereiter | | | X | X |
| Datenschutzbeauftragter | | | X | X |
| DV – Manager | X | | | X |
| Operator | | | X | X |
| Organisator | X | | X | X |
| Revisor | X | X | X | X |
| Sachbearbeiter | | | | X |
| Systemanalytiker | | X | | |
| System – Programmierer | | X | X | |
| Wartungsprogrammierer | | X | X | X |

**Abbildung 8:** Stakeholder nach Scheibl angelehnt an [Sch85]

### 1.8.5. Deutsche Industrienorm

Die Deutsche Industrienorm (DIN) hält fest in der DIN 66230 vom Januar 1981 zur Programmdokumentation, dass diese notwendige Voraussetzung für die Anwendung eines Programmes ist. Daher legt diese DIN den Inhalt der Programmdokumentation fest. Dieser Inhalt lässt sich in zwei Bereiche untergliedern *Benutzerdokumentation* und *Datenverarbeitungstechnische Dokumente*. Die *Benutzerdokumentation* enthält Dokumente, welche der Bereitstellung der erforderlichen Informationen zur Anwendung dienen. Die *Datenverarbeitungstechnische Dokumentation* umfasst Informationen zur Installierung, zum Betrieb und zur Pflege des Programms. Gemäß DIN 66230 [Sch85] lässt sich die gesamte Programmdokumentation mit folgendem Inhaltsverzeichnis darstellen.

1. Programmkenndaten
2. Programmfunktion
3. Programmaufbau
4. Programmablauf
5. Datenorganisation
6. Programmtest
7. Programminstallation
8. Programmbetrieb
9. Anwendungsbeispiel

### 1.8.6. Zusammenfassung und Ergebnis für die PG

Anhand der vorgestellten Beispiele aus der Praxis, wurde im Rahmen der Seminarvorstellung entschieden, welche Dokumente für die Übergabe von PlagTag benötigt werden. Neben den bereits erstellten Dokumenten *Pflichtenheft*, *Entwurfsdokument* und *Testdokument*, werden für die Übergabe weitere Dokumente benötigt. Damit PlagTag in der einzusetzenden Umgebung in Betrieb genommen werden kann, ist ein *Installationshandbuch* zu erstellen. Um die Erweiterbarkeit und Austauschfähigkeit gewährleisten zu können, wird ein *Wartungshandbuch* benötigt, welches es zukünftigen Entwicklern ermöglicht sich leichter in PlagTag einzuarbeiten. Darüber hinaus wird ein *Schulungshandbuch* benötigt, welches dem Zweck der Schulung von Bedienern dienen soll.

Zur Dokumentation des Projektverlaufs sollen folgende Dokumente erstellt werden. Es soll einen *Qualitätsbericht* geben, in dem die erreichte Qualität von PlagTag beschrieben und bewertet wird. Um den Fortschritt und wichtige Meilensteine zu dokumentieren, soll es einen *Fortschrittsbericht* geben. Spezielle für externe Stakeholder soll ein *Abschlussbericht* erstellt werden, in diesem Dokument soll der Projekterfolg und die Zielerreichung sowie der Weg dorthin beschrieben und auf weitere Dokumente verwiesen werden.
Die Dokumentation wurde aufbauend auf den vorgestellten Praxisbeispielen in mehrere Kategorien unterteilt:

- **Projektbegleitende Dokumentation:** Diese umfasst Dokumente, welche bei der Erstellung von PlagTag entstanden sind. Die genauen Dokumente sind das *Exposé*, *Pflichtenheft*, *Entwurf* und *Testdokument*.

- **Anwenderdokumentation:** Diese bezeichnet alle erstellten Dokumente, welche nach der Übergabe helfen PlagTag schnell und einfach in Betrieb nehmen und die Software erweitern oder warten zu können. Diese Dokumente sind *Benutzerhandbuch*, *Installationshandbuch*,

*Schulungshandbuch* und *Wartungshandbuch*.

- **Abschlussdokumentation:** Diese Dokumentation dient der Erfassung der Zielerreichung und gibt einen Überblick über den Verlauf der PG. Dokumente sind *Fortschrittsbericht*, *Qualitätsbericht* und *Abschlussbericht*.

## 1.9. Risikoanalyse

**Autor:** Björn Wolff
Im folgenden Abschnitt werden mögliche Projektrisiken angesprochen und entsprechende Maßnahmen zur Risikoreduktion erläutert.
Die folgenden Punkte wurden als mögliche Risikoquellen identifiziert:

- Zeitplanung
- Ausfall von Teammitgliedern
- Anforderungsänderungen und Change Requests
- Realisierbarkeit der Anforderungen
- Technologieprobleme
- Kompetenzen der Teammitglieder
- Softwarefehler

Bei der *Zeitplanung* kann es schnell zu Engpässen kommen, die durch unvorhergesehene Probleme in der Planung der einzelnen Arbeitsabschnitte entstehen können. Jede der weiteren genannten Risikoquellen hat direkt oder indirekt Einfluss auf die Zeitplanung. Somit ist es notwendig, immer wieder Anpassungen vorzunehmen, die einen reibungslosen Ablauf der Projektgruppe ermöglichen. Zeitengpässe sollen möglichst im Voraus identifiziert und entsprechende Maßnahmen ergriffen werden. Aus diesem Grund bietet sich die Entwicklung in Entwurfszyklen an, die es ermöglicht, je nach zeitlichen Gegebenheiten, Bestandteile der einzelnen Zyklen anzupassen. Zusätzlich sind Teammitglieder für die Zeitplanung zuständig, um die Einhaltung der von der Projektgruppe gesetzten Termine zu überwachen. Die entsprechenden Zuständigkeiten können aus dem Projekthandbuch entnommen werden (siehe Abschnitt 1.7.1). Auch das Nichteinhalten des Zeitplans von einzelnen Teammitgliedern wird, wie im Projekthandbuch angegeben, geahndet.

Der *Ausfall von Teammitgliedern* kann verschiedene Ursachen haben. Dabei kann unterschieden werden zwischen geplanten und ungeplanten Ausfällen. Die geplanten Ausfälle können bei der Zeitplanung berücksichtigt werden, beispielsweise wurde ein Urlaubskalender eingeführt, der die Urlaubstage der Teammitglieder auflistet. Sollte es zu ungeplanten Ausfällen kommen (z.B. durch Krankheit), so muss sichergestellt sein, dass der Aufgabenbereich der ausgefallenen Person ausreichend dokumentiert ist. Das heißt, dass parallel zur Entwicklung bereits dokumentiert werden muss, so dass am Ende eines Entwurfszyklus alles zu der entsprechenden Aufgabe dokumentiert ist. Nur so ist es möglich, dass die anderen Teammitglieder den Ausfall ohne großen Zeitaufwand kompensieren können. Dabei ist auch die Zeitplanung gefragt, die durch großzügiges Zeitmanagement entsprechende Freiräume zulässt. Außerdem müssen Teammitglieder zur Qualitätssicherung eingesetzt werden, die auch überprüfen, ob nach den Richtlinien dokumentiert wurde.

Um besser mit *Anforderungsänderungen und Change Request* umgehen zu können, müssen die einzelnen Anforderungen nachvollziehbar und zwischen den Dokumenten konsistent sein. Das bedeutet,

dass die Herkunft von Anforderungen kenntlich gemacht wird und Querverweise zwischen den Dokumenten ermöglicht werden. So können auch Entscheidungen aus früheren Projektphasen leichter ins Gedächtnis gerufen und eventuelle Change Requests diskutiert werden. Dies erleichtert vor allem die Änderung von Anforderungen in einem späten Projektfortschritt, da hierdurch klar ist, was genau geändert werden muss.

Die *Realisierbarkeit der Anforderungen* sollte bereits früh, durch Erstellung von Prototypen, überprüft werden. Mit Hilfe einer vorläufigen Architektur können erste Tests zur Umsetzbarkeit von Anforderungen gemacht werden, bevor es im späteren Verlauf an diesen Stellen zu Problemen kommt. Auch hier hilft die Entwicklung in Entwurfszyklen, da innerhalb eines jeden Zyklus Tests, sowie eine Analyse und Reflexion stattfindet.

Eine weitere mögliche Risikoquelle sind *Technologieprobleme*. Diese sollen möglichst gering gehalten werden, indem Technologien mit entsprechender Dokumentation genutzt werden. Somit sind viele Fehler, die auftreten können, bereits bekannt und können schnell behoben werden.

Auch die *Kompetenzen der Teammitglieder* sollen beachtet werden, um die Ressourcen möglichst effektiv ausschöpfen zu können. Sollten Sprachen oder Konzepte benutzt werden, die nicht durchgängig bekannt sind, bieten sich Schulungen innerhalb der Projektgruppe an, damit alle über einen ähnlichen Kenntnisstand verfügen. Die Schwerpunkte der einzelnen Teammitglieder sollten darüber hinaus nach Interessen vergeben werden, um die Motivation zu steigern, sich mit den entsprechenden Aufgaben auseinanderzusetzen.

Bei der Entwicklung einer Anwendung ist auch mit *Softwarefehlern* zu rechnen. Um mögliche Fehler schnell beseitigen zu können, wird am Ende eines jeden Zyklus ein Test der gesamten Anwendung durchgeführt. Aber auch während eines Zyklus müssen komponenteninterne Tests durchgeführt werden. Damit ist sichergestellt, dass sich Softwarefehler nicht über die Zyklen hinweg ausbreiten können. Zusätzlich zu den Tests müssen einheitliche Programmierrichtlinien eingehalten werden, welche die Qualität des Programmcodes sichern. Diese Richtlinien werden im Entwurf genauer spezifiziert. Auch hier muss die Qualitätssicherung prüfen, ob alle Richtlinien eingehalten werden.

## 1.10. Überblick

**Autor:** Tore Bierwirth, Sieglinde Hahn
Das Kapitel 2, die *Allgemeine Beschreibung*, dient als Überblick über die zu entwickelnde Anwendung. Hierzu gehört die Beschreibung der Produkt-Einbettung in Form eines Blockdiagramms, sowie der Produkt-Umgebung, welche in einem Domänenmodell dargestellt wird. Anschließend werden allgemeine Anwendungsfälle, sowie die Produkt-Funktionen beschrieben. Gefolgt von den Anforderungen an die Anwender und allgemeinen Rahmenbedingungen. Abgeschlossen wird das Kapitel mit den Annahmen und Abhängigkeiten.

Kapitel 3 beinhaltet die *Spezifischen Anforderungen* und bietet eine detaillierte Beschreibung der Anforderungen, welche für die Entwicklung der Anwendung benötigt werden. Begonnen wird mit einer Definition der rechtlichen Verbindlichkeiten. Nach den funktionalen Anforderungen folgt eine Beschreibung der Leistungsanforderungen, welche an die Anwendung gestellt werden. Gefolgt von den Schnittstellenanforderungen werden anschließend die Datenhaltungsanforderungen beschrieben. Abschließen wird die Beschreibung der *spezifischen Anforderungen* neben den Entwurfs- und Qualitätsanforderungen mit Anforderungen, welche im späteren Verlauf noch geklärt werden müssen.
Im Rahmen der Bestimmung von Anforderungen an PlagTag wurden vier Interviews durchgeführt. Diese Interviews wurden mit Andreas Winter und Jan Jelschen in der Rolle der Auftraggeber geführt. Des Weiteren wurde Thomas Geuken, der Datenschutzbeauftragte der Universität mit dem

Hintergrund interviewt, mögliche rechtliche Konsequenzen erkennen zu können. Ein weiteres Interview fand mit dem Clusterbeauftragten Herrn Reinhard Leidl statt. Die daraus resultierenden Anforderungen sind im Pflichtenheft bei der Anforderungserstellung berücksichtig worden. Die Interviews befinden sich im Anhang (siehe Kapitel E).

# 2. Allgemeine Beschreibung

**Autor:** Marion Gottschalk
PlagTag wird zuerst durch eine *Produkt-Einbettung* beschrieben, um die Komponenten, welche mit PlagTag interagieren, darzustellen. In einer weiteren Ansicht von PlagTag werden die Bestandteile, welche zur Verwendung benötigt werden, durch eine *Produkt-Umgebung* veranschaulicht. Um im weiteren Verlauf die Anforderungen beschreiben zu können, werden die *Rechtlichen Verbindlichkeiten der Anforderungen* erläutert. Daraufhin werden durch ein Anwendungsfalldiagramm die *Allgemeinen Anwendungsfälle* bestimmt. In dem darauffolgenden Kapitel *Produkt-Funktion* werden die Anwendungsfälle durch Aktivitätsdiagramme, konkrete Anwendungsfälle und Mockups beschrieben. Des Weiteren werden unter *Anforderungen an die Anwender*, *Allgemeine Rahmenbedingungen* und *Annahmen und Abhängigkeiten* allgemeine Anforderungen gestellt, welche zur Nutzung der Anwendung wichtig sind.

## 2.1. Produkt-Einbettung

**Autor:** Christoph Gerken, Maxim Klimenko
Das Blockdiagramm in Abbildung 9 gibt einen Überblick über die Umgebung von PlagTag. Der Kern beinhaltet die grundsätzlichen, verarbeitenden Bestandteile. Angesprochen wird PlagTag durch ein Webfrontend, welches der Anwender über einen HTTP-Client (Browser) aufrufen kann. Da PlagTag im Rahmen der Ermittlung möglicher Referenzdaten auf das Internet zugreift, ist eine Anbindung an das Internet nötig. Von hier werden ausschließlich Referenzdaten geladen. Eine Übermittlung detaillierterer Informationen zu einem Plagiatskandidat findet nicht statt. Um große Datenmengen, welche bei der Überprüfung entstehen, bearbeiten zu können, besteht die Möglichkeit den HPC-Cluster der Universität mit einzubeziehen um hierauf große Datenmengen zu bearbeiten. Die Verwendung des Clusters erfordert ein SSH-Verbindung. Im Betrieb wird PlagTag auf einer virtuellen Maschine betrieben werden. Die ermittelten Ergebnisse einer Plagiatsüberprüfung werden persistent abgespeichert.

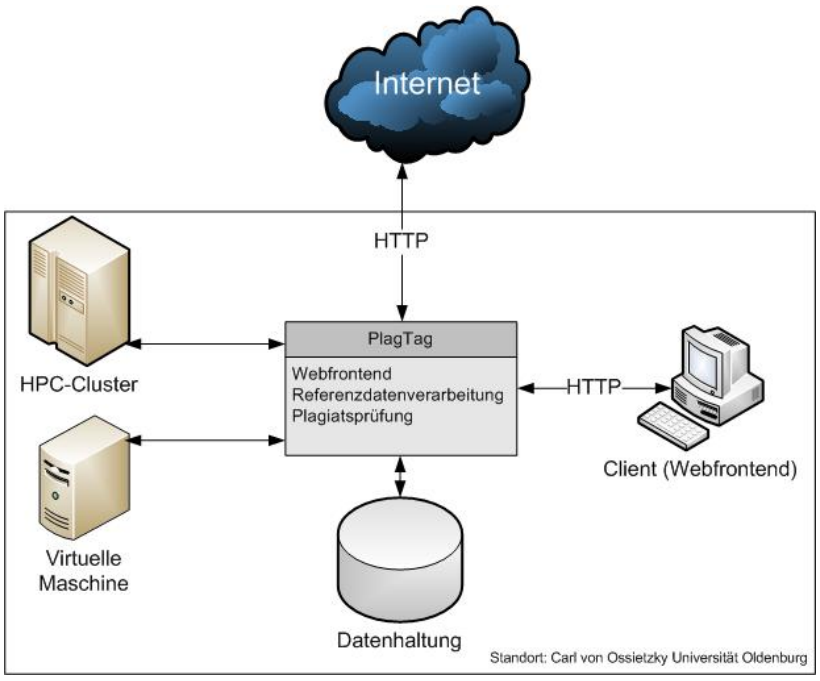


**Abbildung 9:** Blockdiagramm der Anwendung in ihrer Umwelt

Die aus der Betrachtung der Umgebung resultierenden Anwendungs- und Hardware-Schnittstellen werden im folgenden aufgelistet:

- Das Webfrontend wird mittels HTTP über TCP/IP mit dem Client interagieren. Alternativ kann die Verbindung SSL verschlüsselt durchgeführt werden.
- Die Suche nach Referenzdaten im Internet erfolgt mittels HTTP über TCP/IP.
- Für den Client gelten keine direkten Hardwareeinschränkungen, solange einer der unterstützten Browser, wie im Abschnitt 2.6 beschrieben, auf der Hardware funktionsfähig ist und somit auch das Webfrontend genutzt werden kann.
- Als Server-System steht der HPC-Cluster zur Verfügung, welcher in einem späteren Zyklus verwendet werden wird.
- Es erfolgt eine persistente und temporäre Datenhaltung.

## 2.2. Analysemodell

**Autor:** Marion Gottschalk
Die Bestandteile, die von PlagTag zur Plagiatsüberprüfung benötigt werden, werden in diesem Abschnitt durch ein Domänenmodell beschrieben. Dieses ist in Abbildung 10 dargestellt.

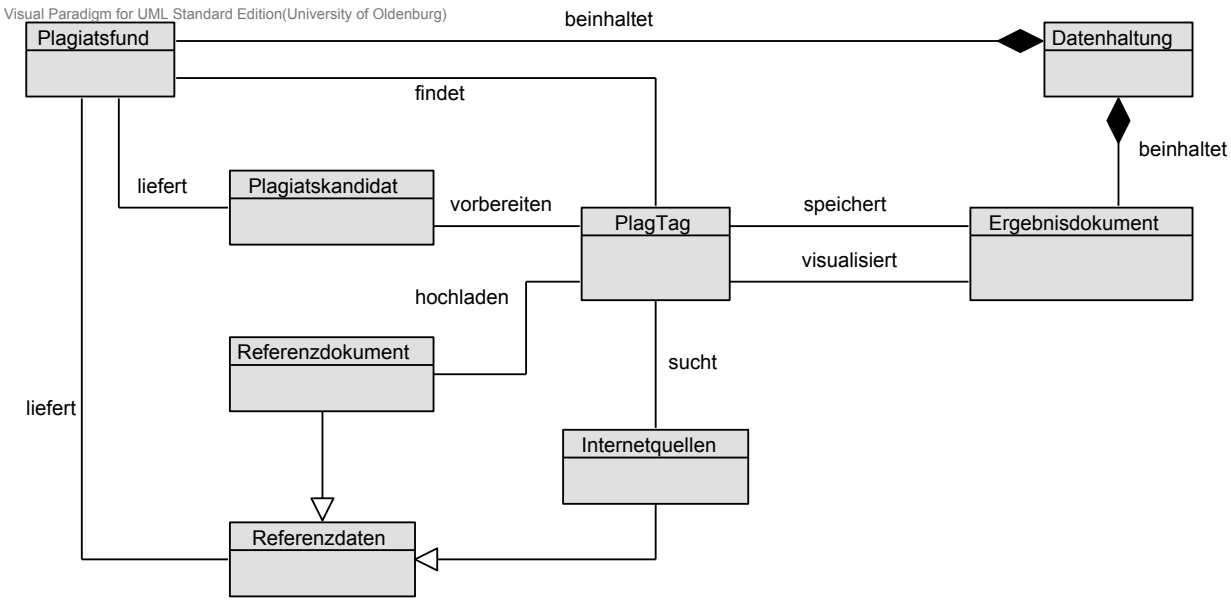


**Abbildung 10:** Domänenmodell der Anwendung

Zur Klärung einiger wichtiger Begriffe sind im Domänenmodell die verschieden Dokumente, welche von PlagTag benötigt bzw. erstellt werden, veranschaulicht. Zuerst wird PlagTag mindestens ein Plagiatskandidat zur Verfügung gestellt werden. Ein Plagiatskandidat ist ein Dokument, welches vom Anwender in PlagTag hochgeladen und auf Plagiate überprüft wird.

Die Plagiatskandidaten werden von PlagTag mit Referenzdaten verglichen. Die Referenzdaten werden in zwei Kategorien unterteilt. Zum einen können vom Anwender selbst beliebig viele Referenzdokumente in PlagTag hochgeladen werden. Die Referenzdokumente sind Dokumente vom Format PDF und TXT, welche direkt mit dem Plagiatskandidat verglichen werden. Zum anderen werden Internetquelle vom Format HTML als Referenzdaten von PlagTag bereitgestellt. Die Internetquelle beinhalten Webseiten, welche voraussichtlich vom Autor plagiiert wurden und werden in Form einer Liste dargestellt, in der entschieden werden kann, ob die Internetquelle weiterhin berücksichtigt

werden soll oder nicht.

Werden beim Vergleich des Plagiatskandidaten mit den Referenzdaten Plagiate gefunden, so werden diese als Plagiatsfund hinterlegt, wobei die Textstelle und die Position im Dokument vom Plagiatskandidat und dem jeweiligen Referenzdaten gespeichert werden. Die Plagiatsfunde werden persistent in der Datenhaltung gespeichert und durch PlagTag im Ergebnisdokument angezeigt.

Pro Plagiatskandidat wird mindestens ein Ergebnisdokument von *PlagTag* erstellt. Das Ergebnisdokument speichert die Informationen aus der Plagiatsüberprüfung des Plagiatskandidaten. Zu den Informationen gehören: *Anzahl der Plagiate, wo wurden die Plagiate im Plagiatskandidaten gefunden* und *welche Referenzdokumente bzw. Internetquelle wurden plagiiert.* Weitere Informationen, welche das Ergebnisdokument beinhalten soll, werden in Kapitel 3.2 beschrieben.

## 2.3. Allgemeine Anwendungsfälle

**Autor:** Marion Gottschalk
Anhand der bisherigen Beschreibung zu PlagTag ergeben sich folgende allgemeinen Anwendungsfälle, welche in Abbildung 11 dargestellt sind. Dieses Anwendungsfalldiagramm soll einen Überblick über den Funktionsumfang von PlagTag geben.

Visual Paradigm for UML Standard Edition(University of Oldenburg)

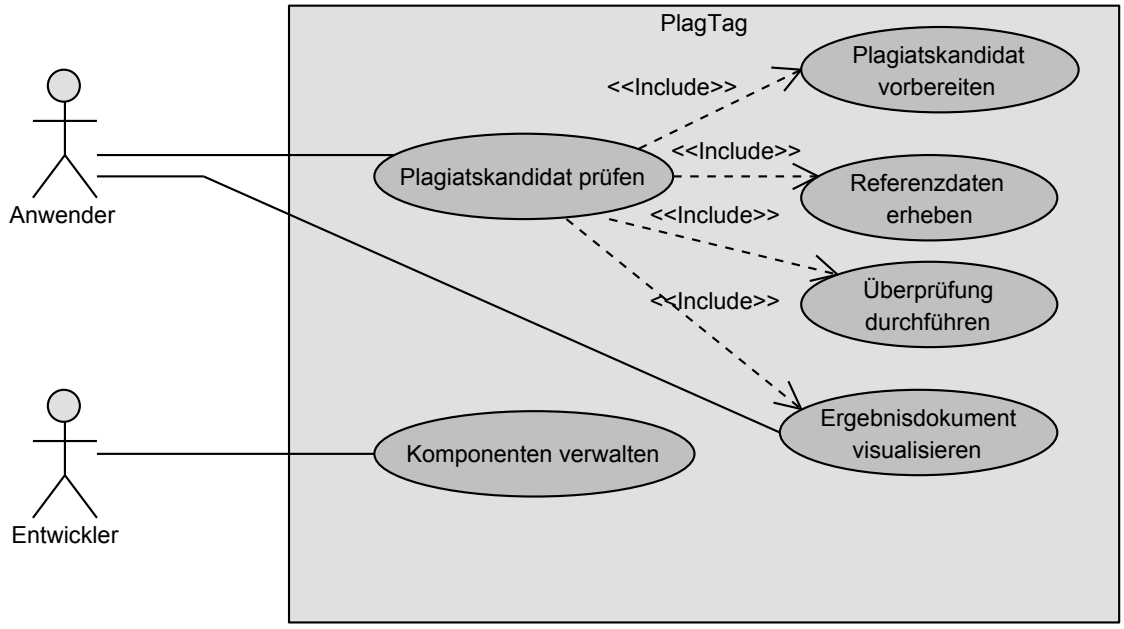


**Abbildung 11:** Anwendungsfalldiagramm zur Anwendung

Der *Anwender* ist derjenige, der PlagTag bedient und einen Nutzen aus den Anwendungsfällen (AF) ziehen soll. Dazu kann der *Anwender* den Anwendungsfall zum *AF1 Plagiatskandidat prüfen* verwenden. *Plagiatskandidat prüfen* beschreibt den übergeordneten Anwendungsfall, welcher aus weiteren Anwendungsfällen besteht. Diese Anwendungsfälle sind *AF1.1 Plagiatskandidaten vorbereiten AF1.2 Referenzdaten erheben, AF1.3 Überprüfung durchführen* und *AF1.4 Ergebnisdokument visualisieren.*

Der Anwendungsfall *AF1.1 Plagiatskandidaten vorbereiten* stellt den Start des *AF1 Plagiatskandidat prüfen* dar und ermöglicht dem Anwender Plagiatskandidaten hinzuzufügen. Einen Nutzen kann der *Anwender* durch die beide Anwendungen *AF1.2 Referenzdaten erheben, AF1.3 Überprüfung durchführen* und *AF1.4 Ergebnisdokument visualisieren* erzielen. Unter *AF1.2 Referenzdaten erheben* hat

der Anwender die Möglichkeit eigene Referenzdaten hochzuladen und nternetquellen selbst auszuwählen. Der Anwendungsfall *AF1.3 Überprüfung durchführen* beschreibt die Plagiatsüberprüfung, welche von PlagTag ausgeführt und vom Anwender eingestellt wird. Unter *AF1.4 Ergebnisdokument visualisieren* können die Ergebnisse von vorherigen Plagiatsüberprüfungen angesehen werden und sind somit für den Anwender abrufbar.

Dem *Entwickler* wird die Möglichkeit geboten, PlagTag durch *AF2 Komponenten verwalten* zu erweitern, aber auch Komponenten zu löschen. Komponenten können in dieser Anwendung z.B. unterschiedliche Algorithmen zur Plagiatsüberprüfung sein. Der Entwickler ist in diesem Fall nicht zwingend die Person, welche die Anwendung zu Beginn erstellt, sondern kann ein Mitarbeiter des Kunden sein, wodurch verstärkt auf die *Wartbarkeit* der Anwendung geachtet werden muss. Die Anforderungen wie z.B. *Wartbarkeit* werden in Kapitel 3 behandelt.

## 2.4. Produkt-Funktion

**Autor:** Marion Gottschalk, Christian Wübbeling
In diesem Abschnitt werden die Funktionen, die PlagTag unterstützen soll, zusammengefasst. Dazu werden *Anwendungsfälle* durch *Aktivitätsdiagramme* dargestellt und daraufhin tabellarisch aufgelistet, um die Anwendung vollständig zu beschreiben. Am Ende werden einige Mockups, die eine mögliches Benutzeroberfläche von PlagTag darstellen, vorgestellt.

Die Auflistung beginnt hier mit einer Übersicht über Anwendungsfälle.

| **Anforderung** | **Anforderungsgeber (Stakeholder)** |
|---|---|
| **AF 1**: Plagiatskandidat prüfen | AW, JJ |
| **AF 1.1**: Plagiatskandidat vorbereiten | AW, JJ, DSB |
| **AF 1.2**: Referenzdaten erheben | AW, JJ, DSB |
| **AF 1.3**: Überprüfung durchführen | AW, JJ, DSB |
| **AF 1.4**: Ergebnisdokument visualisieren | AW, JJ, DSB |
| **AF 2**: Komponenten verwalten | E |

**Tabelle 1:** Übersicht der Anwendungsfälle

**Anforderungsgeber:** Es wurden im Projektverlauf zahlreiche Anforderungsgeber (Stakeholder) identifiziert. Die Wichtigsten sind hier dargestellt:

- Anwender und Kunde: Andreas Winter (AW)
- Anwender: Jan Jelschen (JJ)
- Datenschutzbeauftragter: Thomas Geuken (DSB)
- Entwickler: Jan Jelschen (E)

Die Anwendungsfälle werden im weiteren Verlauf dieses Dokumentes nach folgendem Muster aus Tabelle 2 definiert:

| Feld | Erklärung |
|---|---|
| ID | Anwendungsfälle (AF) werden gemäß dem Anwendungsfalldiagramm nummeriert. |
| Name | Hier folgt eine aussagekräftige Bezeichnung des Anwendungsfalles. |
| Akteure | Hier wird definiert, wer mit dem Anwendungsfall in Kontakt tritt. |
| Verwendete Anwendungsfälle | Hier finden sich andere Anwendungsfälle, die dieser Anwendungsfall verwendet. |
| Auslöser | Wird dieser Anwendungsfall von einem anderen Anwendungsfall aufgerufen, findet sich dies hier wieder. |
| Vorbedingung | An dieser Stelle wird definiert, welche Bedingungen erfüllt sein müssen, damit dieser Anwendungsfall seine Arbeit aufnehmen kann. |
| Nachbedingung Erfolg | Hier steht, was passiert, nachdem dieser Anwendungsfall erfolgreich beendet wurde. |
| Beschreibung | Hier wird beschrieben, was der Anwendungsfall macht. |
| Nachbedingung Fehler / Alternative Abläufe | Sollte es Wahlmöglichkeiten oder spezielle Fehlerbehandlungen geben, wird dies hier definiert. |
| Hinweise | Zusätzliche Hinweise, die in den anderen Feldern nicht gepasst hätten, finden sich hier wieder. |

**Tabelle 2:** Anwendungsfall - Muster

### 2.4.1. Aktivitätsdiagramme und Anwendungsfälle

**Autor:** Tore Bierwirth, Marion Gottschalk, Sieglinde Hahn, Christian Wübbeling

***Plagiatskandidat prüfen***

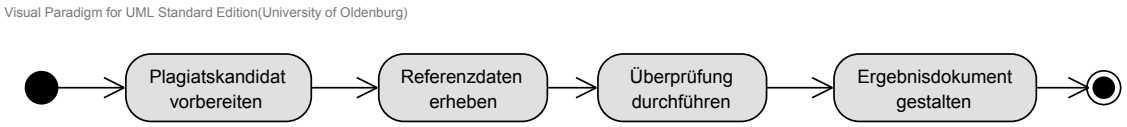


**Abbildung 12:** Verfeinerung des AF1 Plagiatskandidat prüfen

Abbildung 12 und Tabelle 3 stellen die dem Anwendungsfall *AF1 Plagiatskandidat prüfen* zugeordneten Anwendungsfälle *AF1.1 Plagiatskandidat vorbereiten*, *AF1.2 Referenzdaten erheben*, *AF1.3 Überprüfung durchführen* und *AF1.4 Ergebnisdokument gestalten* dar.

| ID | AF1 |
|---|---|
| Name | **Plagiatskandidat prüfen** |
| Akteure | Anwender, PlagTag |
| Verwendete Anwendungsfälle | AF1.1 Plagiatskandidat vorbereiten<br>AF1.2 Referenzdaten erheben<br>AF1.3 Überprüfung durchführen<br>AF1.4 Ergebnisdokument gestalten |
| Auslöser | |
| Vorbedingung | Der Anwender ist in der Anwendung angemeldet und hat den Anwendungsfall aufgerufen. |
| Nachbedingung Erfolg | Referenzdaten werden gem. Datenschutz- und Urheberrecht stets nach Beendigung der Überprüfung gelöscht (DSB). Außerdem wird ein Ergebnisdokument erstellt. |

| ID | AF1 |
|---|---|
| Beschreibung | Der Prozess ist in mehrere Stufen aufgeteilt, die eine Unterbrechung des Prozesses erfordern:<br>1. Vorbereiten von Plagiatskandidaten AF1.1<br>2. Erheben von Referenzdaten AF1.2<br>3. Nach Abschluss beider Anwendungsfälle startet die eigentliche Plagiatsüberprüfung (AF1.3), welche im Erfolgsfall ein Ergebnisdokument erzeugt (AF1.4), welches visualisiert abgerufen werden kann. |
| Nachbedingung Fehler / Alternative Abläufe | Im Fehlerfall (z.B. falls der für die Prüfung zuständige Anwendungsteil aufgrund von Hardware- oder Software-Ausfällen nicht zur Verfügung steht) gilt:<br><br>1) Es wird eine Fehlermeldung ausgegeben bzw. an die aufrufende Komponente zurückgegeben.<br>2a) Steht ein Anwendungsteil nur temporär nicht zur Verfügung, soll der Vorgang später fortgesetzt werden.<br>2b) Steht ein Anwendungsteil dauerhaft nicht zur Verfügung, wird die Prüfung abgebrochen und die Referenzdaten werden gelöscht. |
| Hinweise | Die Prüfung soll unter Zuhilfenahme eines verteilten Systems (HPC-Cluster) erfolgen. |

**Tabelle 3:** AF1 Plagiatskandidat prüfen

Die nachfolgenden Diagramme bieten eine detailliertere Veranschaulichung der in Abbildung 12 dargestellten Aktivitäten. Während sich Abbildung 13 und Tabelle 4 auf *AF1.1 Plagiatskandidat vorbereiten* beziehen, visualisieren Abbildung 14 und Tabelle 5 das Vorgehen zum *AF1.2 Referenzdaten erheben*. *AF1.4 Ergebnisdokument gestalten* wird mithilfe der Abbildung 15 und Tabelle 7 genauer beschrieben. Da *AF1.3 Überprüfung durchführen* eine von *PlagTag* durchgeführte Aktivität ist und nur der Anstoß dieser durch den Anwender geschieht, wird hier auf eine Modellierung durch ein Aktivitätsdiagramm verzichtet. Eine Beschreibung des *AF1.3 Überprüfung durchführen* erfolgt ausschließlich mit Hilfe der Tabelle 6.

### *Plagiatskandidat vorbereiten*

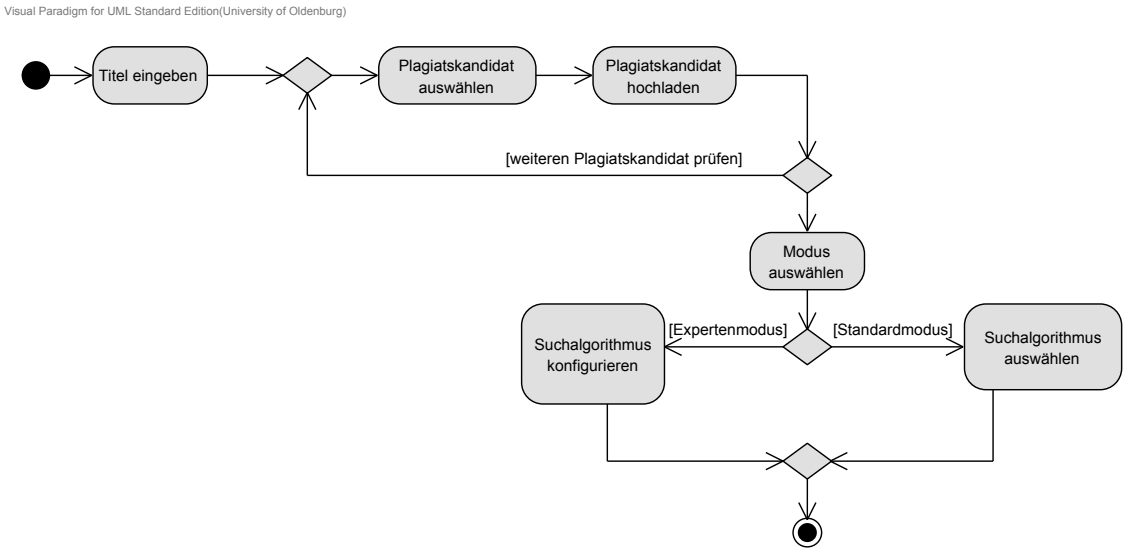


**Abbildung 13:** Plagiatskandidat vorbereiten AF1.1

Dem Interview aus Abschnitt E.2 mit AW ist zu entnehmen, dass PlagTag zwei unterschiedliche Modi bereitstellen soll. Hierzu gehören der *Standardmodus* und der *Expertenmodus*. Während im *Expertenmodus* die Möglichkeit besteht Suchalgorithmen zu konfigurieren, soll im *Standardmodus* lediglich der Suchalgorithmus ausgewählt werden können. Aufgrund dessen, dass der *Expertenmodus* neben den Funktionen, welche auch der *Standardmodus* zur Verfügung stellt, weitere Einstellungsmöglichkeiten bietet, wird sich in Abbildung 13 auf das Vorgehen des *AF1.1 Plagiatskandidat vorbereiten* innerhalb des *Expertenmodus* beschränkt. Ebenso wie im *Standardmodus* ist es hier nach der Eingabe eines Titels möglich, Plagiatskandidaten auszuwählen und die entsprechenden Dokumente hochzuladen. Alle Dokumente, die unter dem gleichen Titel hochgeladen wurden, werden zu einer Überprüfung zusammengefasst, in welcher der Titel als Überprüfungsname fungiert. Zu den spezifischen Einstellungen, welche laut dem Interview innerhalb des *Expertenmodus* zu berücksichtigen sind, gehört die Wahl eines Suchalgorithmus. Alle getroffenen Einstellungen werden für die gesamte Gruppe übernommen. Weitere Erläuterungen zu diesem Anwendungsfall finden sich in Tabelle 4.

| ID | AF1.1 |
|---|---|
| Name | **Plagiatskandidat vorbereiten** |
| Akteure | Anwender |
| Verwendete Anwendungsfälle | AF1.2: Referenzdaten erheben |
| Auslöser | AF1: Plagiatskandidat prüfen |
| Vorbedingung | Anwender ist in PlagTag angemeldet und hat AF1 aufgerufen. |
| Nachbedingung Erfolg | Die hochzuladenden Dokumente werden verarbeitet, indexiert und temporär in der Datenhaltungskomponente abgelegt. Nach Ende der Verarbeitung ändert sich der Auftragsstatus und der Anwender kann im Erfolgsfall den Prozess mit AF1.2 fortsetzen. |
| Beschreibung | Der Anwender kann zu prüfende Dokumente hochladen.<br>Diese werden im Anschluss im Hintergrund indexiert, temporär gespeichert und im Hinblick auf Literaturvorschläge verarbeitet sowie eine Suche nach Internetquellen gestartet. |
| Nachbedingung Fehler / Alternative Abläufe | Können einzelne Dokumente nicht verarbeitet werden, wird dies dem Anwender unter Angabe des Dateinamens und des Fehlergrundes ausgegeben. Er erhält die Möglichkeit, erneut Referenzdokumente hochzuladen. |
| Hinweise | Die hochzuladenden Dokumente müssen im Format PDF oder TXT geliefert werden. |

**Tabelle 4:** AF1.1 Plagiatskandiadaten vorbereiten

### *Referenzdaten erheben*

Visual Paradigm for UML Standard Edition(University of Oldenburg)

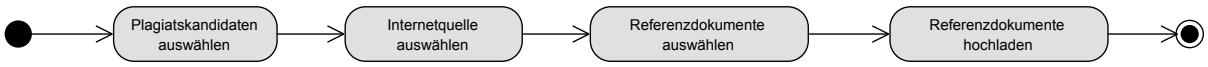


**Abbildung 14:** Referenzdaten erheben AF1.2

Abbildung 14 und Tabelle 5 beschreiben den Vorgang *AF1.2 Referenzdaten erheben*. Um gezielter nach Plagiaten suchen zu können wird es auf Wunsch des Interviewpartners AW möglich sein, aus einer Auswahl an vorgeschlagenen Internetquellen zu selektieren, welche für die Überprüfung verwendet werden sollen. Einen weiteren Grund für die Möglichkeit der Selektion von Internetquellen lässt sich aus dem Interview mit dem DSB ableiten. Da die Verwendung illegal erworbener

Referenzdaten unzulässig ist, können über diese Option fragwürdige Internetquellen für die Überprüfung ausgeschlossen werden. Bei einer ausschließlichen Nutzung von Internetquellen, wird eine Abwahl aller Internetquellen dazu führen, dass plagiierte Textpassagen nicht erkannt werden. Aus diesem Grund wird dem Anwender neben dem Selektieren der bereits genannten Internetquellen zusätzlich die Möglichkeit geboten eigene Referenzdokumente auszuwählen und hochzuladen, um diese PlagTag für die Überprüfung zur Verfügung zu stellen. Eine Anbindung an elektronische Datenbanken wie Springer oder IEEE zum automatischen Herunterladen von PDFs, welche dann als Referenzdokumente verwendet werden können, wird aus rechtlichen Gründen nicht realisiert.

| ID | AF1.2 |
|---|---|
| Name | **Referenzdaten erheben** |
| Akteure | Anwender |
| Verwendete Anwendungsfälle | AF1.1: Plagiatskandidat vorbereiten |
| Auslöser | AF1.1: Plagiatskandidat vorbereiten |
| Vorbedingung | Die Verarbeitung von AF1.1: (Plagiatskandidat vorbereiten) ist abgeschlossen |
| Nachbedingung Erfolg | Die nächste Phase der Plagiatsüberprüfung wird gestartet (AF1). |
| Beschreibung | Der Anwender hat im Falle von Internetquellen die Möglichkeit, das Durchsuchen von (z.B. offensichtlich urheberrechtlich fragwürdigen Quellen) einzeln pro Quelle zu deaktivieren (DSB).<br>Er kann weiterhin in der erstellten Liste der Literatur-Vorschläge gezielt Referenzdokument-Vorschläge ähnlich einer Merkliste streichen und Referenzdokumente hochladen.<br>Der Anwender kann weiterhin einige noch zu definierende Prüfoptionen festlegen. |
| Nachbedingung Fehler / Alternative Abläufe | Können einzelne Referenzdokumente nicht verarbeitet werden, wird dies dem Anwender unter Angabe des Dateinamens und des Fehlergrundes ausgegeben. Er erhält die Möglichkeit, erneut Referenzdokumente hochzuladen. |
| Hinweise | Die hochzuladenden Dokumente müssen im Format PDF oder TXT geliefert werden. |

**Tabelle 5:** AF1.2 Referenzdaten erheben

### *Überprüfung durchführen*

Mit Hilfe der Tabelle 6 wird der Anwendungsfall *AF1.3 Überprüfung durchführen* genauer beschrieben.

| ID | AF1.3 |
|---|---|
| Name | **Überprüfung durchführen** |
| Akteure | Anwender |
| Verwendete Anwendungsfälle | |
| Auslöser | AF1.2: Referenzdaten erheben |
| Vorbedingung | Die Verarbeitung von AF1.2: (Referenzdaten erheben) ist abgeschlossen |
| Nachbedingung Erfolg | Die Überprüfung der Plagiatskandidaten wurden abgeschlossen. Das Ergebnisdokument der Überprüfung wird erzeugt und kann zur Betrachtung ausgewählt werden. |
| Beschreibung | PlagTag überprüft auf Grundlage der ausgewählten Referenzdaten und der ausgewählten Überprüfungsoptionen (Suchalgorithmus etc.) die gewählten Plagiatskandidaten. Eine Unterbrechung einer laufenden Suche ist nicht möglich. |

| ID | AF1.3 |
|---|---|
| | Suchalgorithmen sollen über eine komponentenbasierte Architektur austauschbar sein (siehe AF2). Es stehen nur diejenigen Algorithmen zur Verarbeitung zur Verfügung, die in AF2 eingebunden wurden. |
| Nachbedingung Fehler / Alternative Abläufe | Im Fehlerfall (z.B. falls der für die Prüfung zuständige Anwendungsteil z.B. aufgrund von Hardware- oder Software-Ausfällen nicht zur Verfügung steht) gilt:<br>Es wird eine Fehlermeldung ausgegeben bzw. an die aufrufende Komponente zurückgegeben. |
| Hinweise | |

**Tabelle 6:** AF1.3 Überprüfung durchführen

***Ergebnisdokument gestalten***

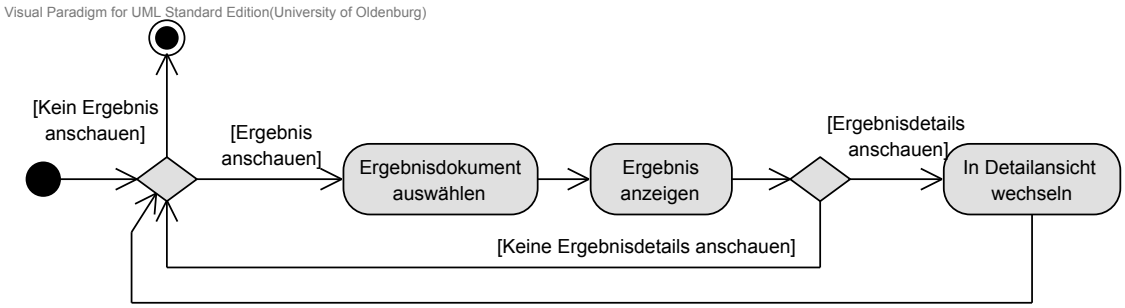


**Abbildung 15:** Ergebnisdokument gestalten AF1.4

Das Aktivitätsdiagramm der Abbildung 15 und Tabelle 7 beschreiben die Vorgehensweise *AF1.4 Ergebnisdokument gestalten* für Dokumente, deren Überprüfung bereits abgeschlossen ist. Nach der Auswahl des entsprechenden Ergebnisdokuments kann der Anwender sich das Ergebnis der Überprüfung anzeigen lassen. Neben der normalen Ergebnisausgabe ist es möglich, in eine Ansicht zu wechseln, in welcher die Ergebnisse detaillierter dargestellt werden. Die Detailansicht beinhaltet auf Wunsch von AW unter Anderem eine Gegenüberstellung des Originaltext mit dem möglichen Plagiat.

| ID | AF1.4 |
|---|---|
| Name | **Ergebnisdokument gestalten** |
| Akteure | Anwender |
| Verwendete Anwendungsfälle | AF1: Plagiatskandidat prüfen |
| Auslöser | AF1.2: Referenzdaten erheben |
| Vorbedingung | Die Plagiatsüberprüfung wurde abgeschlossen. |
| Nachbedingung Erfolg | |
| Beschreibung | Das durch AF1 generierte Ergebnisdokument wird dem Anwender ausgegeben. Aus dem Verhältnis von Text-Anteilen mit Plagiatsverdacht zu Text-Anteilen ohne Fund soll ein Balken-Diagramm ähnlich guttenplag-Blog und ein prozentualer/farblicher Indikator errechnet werden (siehe Mockups 19 und 20). Der Bericht muss die Verdachts-Stellen den Funden samt Quellen gegenüberstellen sowie Kapitel und etwas umliegenden Text darstellen. Aussagen sollen stets vage gehalten werden, da Prüfer die Verdachtsfälle verifizieren müssen. |

| ID | AF1.4 |
|---|---|
| Nachbedingung Fehler / Alternative Abläufe | Falls das gewünschte Dokument nicht vorliegt oder nicht erstellt werden kann, wird eine Fehlermeldung ausgegeben. Der Anwender erhält ggf. die Möglichkeit, einen neuen Prüfprozess in AF1 zu starten. |
| Hinweise | |

**Tabelle 7:** AF1.4 Ergebnisdokument gestalten

### 2.4.2. Mockups

**Autor:** Christoph Gerken

Dieses Kapitel zeigt mögliche grafische Darstellungen der Benutzeroberfläche von PlagTag. Die entwickelten Darstellungen sollen im Rahmen der Anforderungsermittlung als Hilfestellung dienen, um die Anforderungen an die grafische Benutzeroberfläche zu erkennen und zu prüfen. In den im folgenden beschriebenen Mockups sind Entwürfe für das Webfrontend der Software abgebildet. Es handelt sich hierbei nicht um endgültige Aussagen zum Erscheinungsbild der Software.

In Abbildung 16 ist die Startseite des Webfrontends als Mockup zu erkennen. Diese Seite erscheint nach der Anmeldung in das Plagiatserkennungssystem. Wie in den durchgeführten Interviews mit AW und JJ festgehalten wurde (siehe Anhang E), kann der Anwender im oberen Bereich den Status seiner gegenwärtigen Überprüfungen sehen. Je nach Fortschritt sind hier diverse Detailanzeigen möglich. Diese lassen sich durch einen Klick ansehen bzw. bearbeiten.

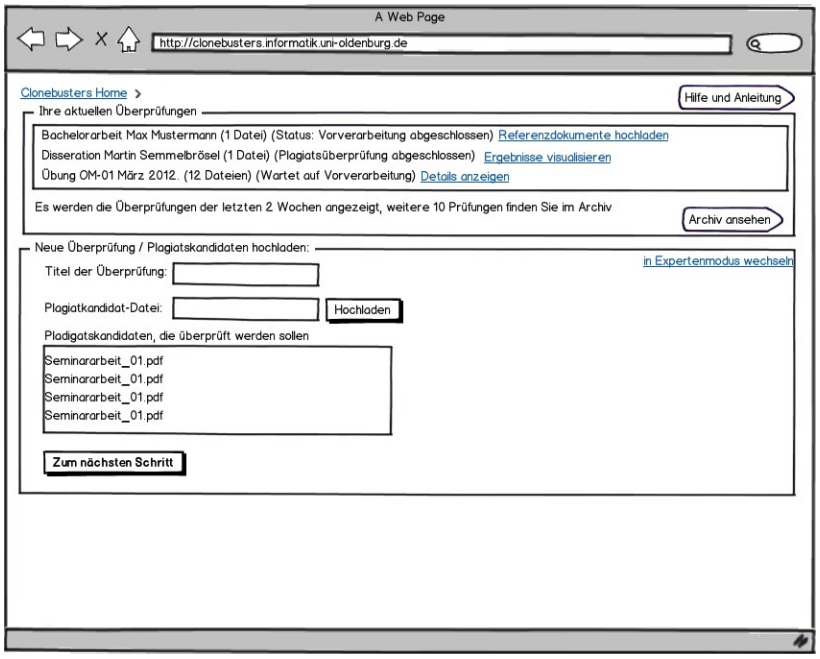
A Web Page
http://clonebusters.informatik.uni-oldenburg.de
Clonebusters Home
Hilfe und Anleitung
Ihre aktuellen Überprüfungen
Bachelorarbeit Max Mustermann (1 Datei) (Status: Vorverarbeitung abgeschlossen) Referenzdokumente hochladen
Disseration Martin Semmelbrösel (1 Datei) (Plagiatsüberprüfung abgeschlossen) Ergebnisse visualisieren
Übung OM-01 März 2012. (12 Dateien) (Wartet auf Vorverarbeitung) Details anzeigen
Es werden die Überprüfungen der letzten 2 Wochen angezeigt, weitere 10 Prüfungen finden Sie im Archiv
Archiv ansehen
Neue Überprüfung / Plagiatskandidaten hochladen:
in Expertenmodus wechseln
Titel der Überprüfung:
Plagiatkandidat-Datei:
Hochladen
Pladigatskandidaten, die überprüft werden sollen
Seminararbeit_01.pdf
Seminararbeit_01.pdf
Seminararbeit_01.pdf
Seminararbeit_01.pdf
Zum nächsten Schritt

**Abbildung 16:** Mockup der Startseite - Standardmodus

Um die Benutzerfreundlichkeit zu erhöhen, lässt sich im selben Fenster direkt eine neue Überprüfung starten. Hierzu muss der Titel der Überprüfung angegeben werden. Der Titel ist der eindeutige Namensgeber für die Überprüfung. Dies ist sinnvoll, damit sich die Überprüfungen gruppieren lassen. Ein möglicher Anwendungsfall könnte sein, dass eine Gruppe von Seminararbeiten gemeinsam überprüft werden kann und diese Überprüfung dann als Gruppe verarbeitet wird. Einem Titel können mehrere Plagiatskandidaten zugeordnet werden. Im nächsten Feld können dann Plagiatskandidaten zur Überprüfung ausgewählt werden. Eine Überprüfung kann parallel mehrere Dateien verarbeiten. Die bereits zur Verarbeitung ausgewählten Plagiatskandidaten werden unter dem Dateifeld aufge-

listet. Wie in den Interviews mit AW und JJ festgestellt wurde (siehe Anhang E), benötigt PlagTag eine Oberfläche die sich an den Fähigkeiten des Anwenders orientiert. Der Anwender kann deshalb in dieser Ansicht zwischen dem Expertenmodus und dem Standardmodus wählen. Grundsätzlich ist der Standardmodus vorausgewählt. Der Expertenmodus wird in Abbildung 17 visualisiert.
Sind alle Dateien hochgeladen, kann die Software gestartet werden. Daraufhin wird eine Vorverarbeitung durchgeführt. In dieser Vorverarbeitung werden die Plagiatskandidaten aufbereitet.

In Abbildung 17 ist der Expertenmodus zu sehen, das Hinzufügen der Dateien erfolgt wie im Standardmodus, zusätzlich können weitere Parameter eingegeben werden, zum Beispiel der zu verwendende Suchalgorithmus und die Suchtiefe (in Abhängigkeit der gewählten Algorithmen). Diese Optionen wurden im Rahmen der durchgeführten Interviews (siehe Anhang E) als relevante Optionen für den Expertenmodus ermittelt. Im nächsten Schritt erfolgt dann ebenfalls die Vorverarbeitung. Der nächste Schritt entspricht demselben Schritt wie dem im Standardmodus.

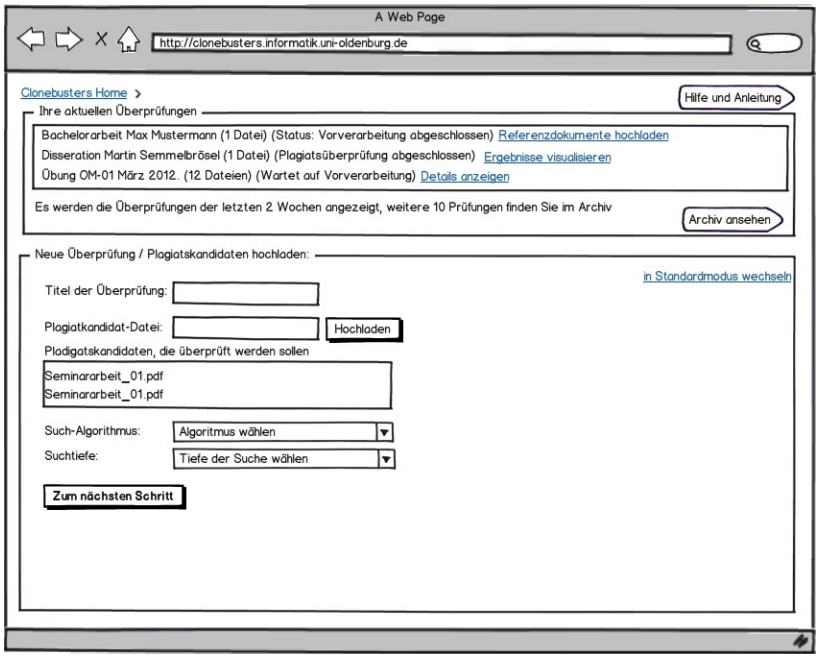
A Web Page
http://clonebusters.informatik.uni-oldenburg.de
Clonebusters Home
Hilfe und Anleitung
Ihre aktuellen Überprüfungen
Bachelorarbeit Max Mustermann (1 Datei) (Status: Vorverarbeitung abgeschlossen) Referenzdokumente hochladen
Disseration Martin Semmelbrösel (1 Datei) (Plagiatsüberprüfung abgeschlossen) Ergebnisse visualisieren
Übung OM-01 März 2012. (12 Dateien) (Wartet auf Vorverarbeitung) Details anzeigen
Es werden die Überprüfungen der letzten 2 Wochen angezeigt, weitere 10 Prüfungen finden Sie im Archiv
Archiv ansehen
Neue Überprüfung / Plagiatskandidaten hochladen:
in Standardmodus wechseln
Titel der Überprüfung:
Plagiatkandidat-Datei:
Hochladen
Pladigatskandidaten, die überprüft werden sollen
Seminararbeit_01.pdf
Seminararbeit_01.pdf
Such-Algorithmus:
Algoritmus wählen
Suchtiefe:
Tiefe der Suche wählen
Zum nächsten Schritt

**Abbildung 17:** Mockup der Startseite - erweiterter Eingabemodus

Abbildung 18 zeigt Optionen an, welche im Rahmen der Vorverarbeitung ermittelt wurden und lässt diese vom Anwender verifizieren. Die Vorverarbeitung teilt sich in drei Bereiche auf. Im oberen Bereich werden Internetquellen angezeigt, die für die Überprüfung relevant sein können. Wie in den Interviews festgehalten, besteht die Option diese vor der Überprüfung zu selektieren, um auszuschließen, dass es sich hierbei um illegal verfügbare Inhalte handelt.
Des Weiteren könnte PlagTag Quellen anzeigen, welche vermutlich für die Plagiatssuche relevant sind, aber nicht zu Verfügung stehen. Der Anwender hat dann die Möglichkeit, diese Quellen der Anwendung zur Verfügung zu stellen. Als weiteren Punkt bietet PlagTag die Möglichkeit eigene Referenzdokumente hinzuzufügen. Wurden die Eingaben erfolgreich durchgeführt, wird die Prüfung gestartet.

Abbildung 19 zeigt eine Übersichtsseite zur Auswertung der Ergebnisse. Im oberen Bereich können die zur Überprüfung eingereichten Plagiatskandidaten ausgewählt werden. Ein Klick auf *Auswertung anzeigen* zeigt dann das Dashboard zum jeweiligen Plagiatskandidat an. Das Dashboard besteht aus vier Ansichten, zu denen jeweils Details aufgerufen werden können. Ein Kuchendiagramm zeigt die prozentuale Menge der gefunden Plagiate im Verhältnis zum gesamten Text an. Eine zweite

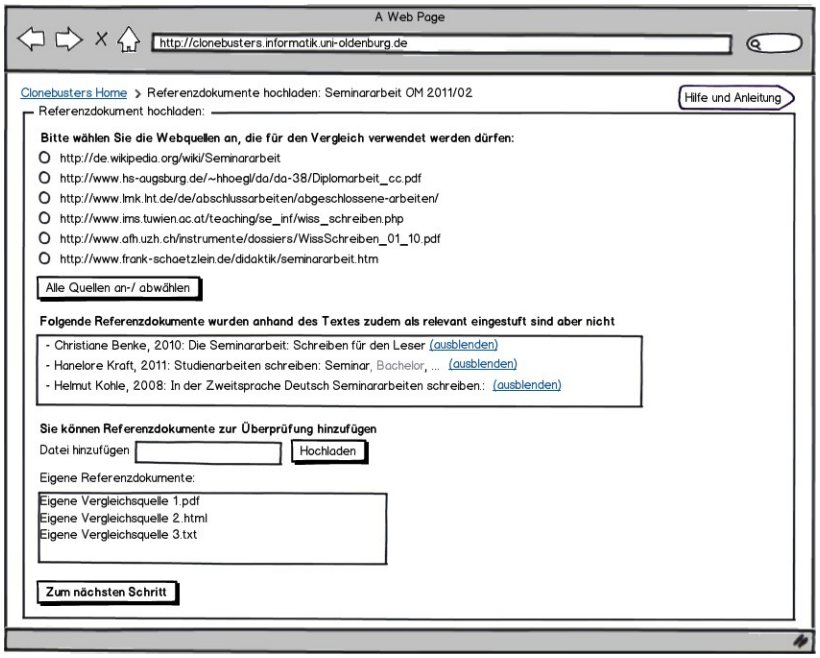
A Web Page
http://clonebusters.informatik.uni-oldenburg.de
Clonebusters Home > Referenzdokumente hochladen: Seminararbeit OM 2011/02
Hilfe und Anleitung
Referenzdokument hochladen:
Bitte wählen Sie die Webquellen an, die für den Vergleich verwendet werden dürfen:
http://de.wikipedia.org/wiki/Seminararbeit
http://www.hs-augsburg.de/~hhoegl/da/da-38/Diplomarbeit_cc.pdf
http://www.lmk.lnt.de/de/abschlussarbeiten/abgeschlossene-arbeiten/
http://www.ims.tuwien.ac.at/teaching/se_inf/wiss_schreiben.php
http://www.afh.uzh.ch/instrumente/dossiers/WissSchreiben_01_10.pdf
http://www.frank-schaetzlein.de/didaktik/seminararbeit.htm
Alle Quellen an-/ abwählen
Folgende Referenzdokumente wurden anhand des Textes zudem als relevant eingestuft sind aber nicht
- Christiane Benke, 2010: Die Seminararbeit: Schreiben für den Leser (ausblenden)
- Hanelore Kraft, 2011: Studienarbeiten schreiben: Seminar, Bachelor, ... (ausblenden)
- Helmut Kohle, 2008: In der Zweitsprache Deutsch Seminararbeiten schreiben.: (ausblenden)
Sie können Referenzdokumente zur Überprüfung hinzufügen
Datei hinzufügen
Hochladen
Eigene Referenzdokumente:
Eigene Vergleichsquelle 1.pdf
Eigene Vergleichsquelle 2.html
Eigene Vergleichsquelle 3.txt
Zum nächsten Schritt

**Abbildung 18:** Mockup der Ansicht: Referenzdokumente hochladen

Übersicht zeigt die Dokumente an, die am häufigsten plagiiert wurden. Der Strichcode gibt an, bei welchen Stellen im Text Plagiate erkannt wurden. Die vierte Ansicht gibt Aufschluss über die von der Anwendung korrekt erkannten zitierte Textpassagen im Verhältnis zu gefundenen Plagiaten. Diese Ansichten können parametrisiert und detailliert betrachtet werden.

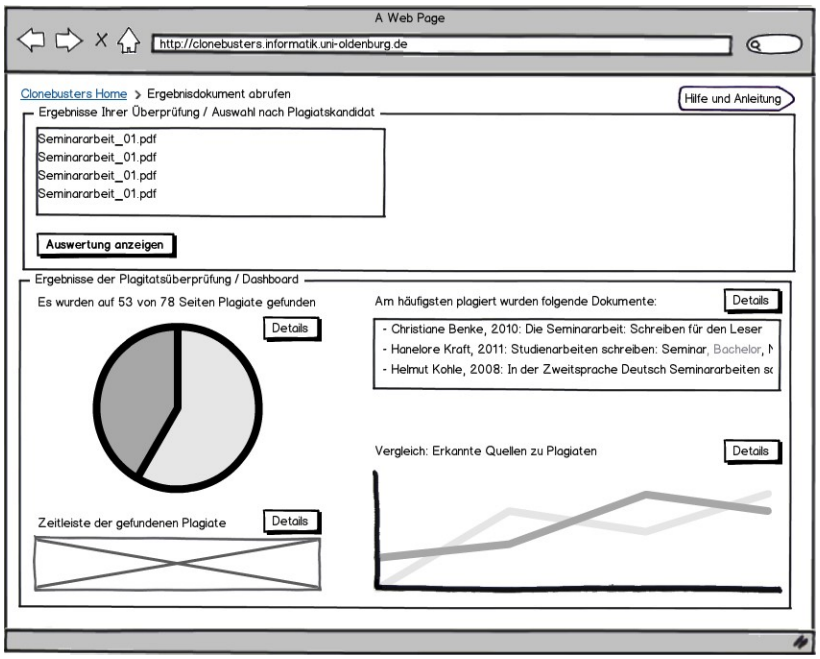


**Abbildung 19:** Mockup der Seite: Ergebnisdokument visualisieren

In Abbildung 20 ist der textuelle Vergleich zu sehen, der die Auswertung eines Plagiatskandidaten mit einem Referenzdokument detailliert gegenüberstellt. Der Bereich der Zeitleiste stellt die komplette Länge des Dokuments dar. Farbliche Hervorhebungen zeigen die Text-Passagen mit möglichen Plagiaten auf. Über die als Schieberegler funktionierende Strichcode kann zur jeweiligen Position im Text gesprungen werden.

Im unteren Bereich des Webfrontends, siehe Abbildung 20 wird dann links der passend hinterlegte Text-Abschnitt angezeigt. Parallel zu diesem lässt sich die Text-Passage des Referenzdokuments auf der rechten Seite einblenden. Sollten mehrere Quellen als Ursprung des Textes erkannt worden sein, so lassen sich diese über eine Liste auswählen.

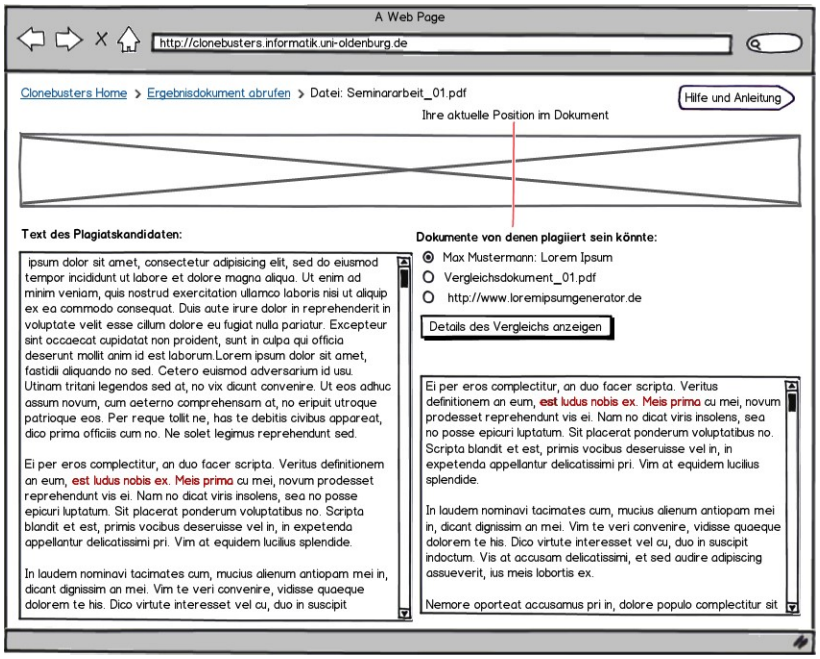


**Abbildung 20:** Mockup der Seite: Ergebnisdokument visualisieren - Detailansicht des Strichcode

## 2.5. Anforderung an Benutzerfreundlichkeit

**Autor:** Björn Wolff

Das Webfrontend von PlagTag wird einfach zu bedienen sein, so dass selbst Anwender, die noch nie mit einer anderen Plagiatssoftware gearbeitet haben, PlagTag benutzen können. Dies wird an Hand eines, im späteren Verlauf des Projekts folgenden, Usability-Tests überprüft. Das Webfrontend ist hierbei gleichzusetzen mit einer grafischen Benutzeroberfläche für den Anwender. Zur Illustration wurden bereits in Kapitel 2.4.2 einige Mockups dargestellt. Hierbei werden die unterschiedlichen Verarbeitungsschritte der Plagiatssuche auf verschiedenen Seiten in verständlicher Form dargestellt. Sollte der Anwender weitere Einstellungen vornehmen wollen, die über die einfache Suche nach Plagiaten hinausgehen, so kann er den Expertenmodus wählen. Dieser Modus ermöglicht es, mehr Einfluss auf die Art und Weise der Plagiatssuche zu haben. Zum leichteren Einstieg in die Benutzung des Webfrontends wird es ein Handbuch mit typischen Anwendungsbeispielen geben.
Für Entwickler wird es zusätzlich eine Dokumentation des Programmcodes geben, sowie Beschreibungen zur Entwicklung und Integration neuer Module für PlagTag. Dies soll dazu beitragen, dass eine Weiterentwicklung leicht umzusetzen ist.

## 2.6. Allgemeine Rahmenbedingungen

**Autor:** Björn Wolff

Für den Anwender wird ein Webfrontend zur Verfügung gestellt, weshalb lediglich ein Browser mit Internetzugriff erforderlich ist. Dieses umfasst das Vorhandensein der in Abschnitt 2.1 beschriebenen Kommunikationsprotokolle. Das Webfrontend soll mindestens mit folgenden Browsern genutzt werden können:

- Mozilla Firefox Version 8
- Google Chrome Version 15
- Apple Safari Version 5

Die Ausstattung des zu benutzenden Servers muss bei der Entwicklung von PlagTag berücksichtigt werden. Die Auswahl der Entwicklungssprache wird bei der Entscheidung über die zu nutzende Architektur getroffen.

## 2.7. Annahmen und Abhängigkeiten

**Autor:** Björn Wolff
Der wesentliche Teil von PlagTag wird auf einem Server betrieben und ist daher auch hauptsächlich von diesem abhängig. Sollten also Änderungen am Server vorgenommen werden, sei es beispielsweise durch Updates oder Betriebssystemwechsel, so muss sichergestellt werden, dass diese mit dem darauf ausgeführten PlagTag kompatibel ist. Daher ist eine Abstimmung mit dem Administrator des Servers notwendig, um mögliche Versionskonflikte zwischen Server und PlagTag zu identifizieren und gegebenenfalls anzupassen.
Nach außen existiert nur der Zugriff auf das Webfrontend, welches durch Änderungen an der Systemumgebung nicht betroffen ist.

# 3. Spezifische Anforderungen

**Autor:** Tore Bierwirth
Innerhalb der *spezifischen Anforderungen* werden zu Beginn die *rechtlichen Verbindlichkeiten* und die *funktionalen Anforderungen* definiert. Der darauf folgende Abschnitt der *Leistungsanforderungen* definiert Richtwerte, welche PlagTag einhalten muss. Anschließend werden *Entwurfs- und Schnittstellenanforderungen* an PlagTag gestellt. Hier wird aufgelistet, welche Anforderungen an den Systementwurf gestellt werden. Wie Daten für PlagTag abgespeichert werden sollen wird in den *Datenhaltungsanforderungen* beschrieben. Abgeschlossen werden die *spezifischen Anforderungen* mit den *Qualitätsanforderungen*, welche an PlagTag gestellt werden. Hierbei werden auf nichtfunktionale Anforderungen wie beispielsweise die Zuverlässigkeit sowie die Sicherheit eingegangen.
Hier wird zudem eine einheitliche Notation für die Anforderungsschlüssel eingeführt. Der Schlüssel besteht aus einem Buchstaben gefolgt von Zahlen. Der Buchstabe soll Aufschluss darüber geben, welchem Bereich die Anforderung zuzuordnen ist. Die Kombination aus Buchstabe und Ziffernfolge ermöglicht eine eindeutige Identifikation der Anforderung. Die Klammer am Ende jeder Anforderung stellt einen Verweis zur Herkunft der Anforderung dar. Dadurch soll das Zurückverfolgen der Anforderungen zum entsprechenden Kapitel vereinfacht werden.

## 3.1. Rechtliche Verbindlichkeit der Anforderungen

**Autor:** Marion Gottschalk, Christian Wübbeling
Die herausgestellten Anforderungen werden nach folgender rechtlicher Verbindlichkeit behandelt.

- Pflichtanforderungen (Muss): Diese Anforderungen müssen realisiert werden, um die Grundanforderungen des Kundens zu erfüllen und damit den Vertrag rechtlich zu erfüllen.
- Wunschanforderungen (Sollte): Diese Anforderungen sind für die Anwendung optional, sollten aber bei Möglichkeit realisiert werden und dies vor den *Vorschlagsanforderungen*.
- Vorschlagsanforderungen (Kann): Diese Anforderung sind für die Anwendung optional.

## 3.2. Funktionale Anforderungen

**Autor:** Tore Bierwirth, Sieglinde Hahn, Maxim Klimenko, Björn Wolff
Die *funktionalen Anforderungen* enthalten die Anforderungen, die im Verlauf des Projektes identifiziert worden sind

- *F010* Es müssen textuelle Plagiate erkannt werden (1.4).
- *F020* Komplettplagiate müssen erkannt werden (1.5).
- *F030* Verschleierungen müssen erkannt werden (1.5).
- *F040* Verschleierungen mit 0% wörtlicher Übereinstimmung dürfen nicht erkannt werden (1.5).
- *F050* Halbsatzflickereien sollten erkannt werden (1.5).
- *F060* Shake and Pastes sollten erkannt werden (1.5).
- *F070* Alibi-Fußnoten sollten erkannt werden (1.5).
- *F080* Strukturplagiate sollten erkannt werden (1.5).
- *F090* Eigenplagiate sollten erkannt werden (1.5).
- *F100* Kopierte Zitate können erkannt werden (1.5).

- *F110* Die Erkennung der Parodie kann wegen der Seltenheit in wissenschaftlichen Arbeiten ausgeschlossen werden (1.5).
- *F120* Plagiatskandidaten müssen mit Referenzdaten verglichen werden (2.2).
- *F130* Für einen Plagiatskandidaten muss mindestens ein Ergebnisdokument erstellt werden (2.2).
- *F140* Internetquellen müssen automatisch angeboten werden (2.2).
- *F150* Internetquellen müssen selektierbar sein (2.2).
- *F160* Das Ergebnisdokument muss die plagiierten Textpassagen enthalten (2.2).
- *F170* Das Ergebnisdokument muss Informationen darüber beinhalten, wo die Plagiate im Plagiatskandidaten gefunden wurden (2.2).
- *F180* Auf plagiierten Referenzdaten muss im Ergebnisdokument verwiesen werden (2.2).
- *F190* PlagTag muss automatisch die verwendete Sprache des Plagiatskandidaten erkennen (1.6).
- *F200* PlagTag muss die Möglichkeit bieten manuell die Sprache festzulegen (1.6).
- *F210* Der Status der Verarbeitung muss angezeigt werden können (1.6).
- *F220* Gesamtübersicht der gefundenen Plagiate muss in Form eines Kuchendiagramm visualisiert werden (1.6).
- *F230* Es muss eine textuelle Gegenüberstellung von Referenzdaten und Plagiatskandidaten geben (1.6).
- *F240* Referenzdaten müssen bei der Visualisierung selektierbar sein (1.6).
- *F250* PlagTag kann eine Abschätzung abliefern, welche Plagiatskandidaten einer Plagiatsüberprüfung Plagiate enthalten könnten (1.6).
- *F260* PlagTag kann mögliche Plagiatskandidaten farblich hervorheben (1.6).
- *F270* Komplettplagiate und Verschleierungen können farblich unterschieden werden (1.6).
- *F280* Das Ergebnisdokument muss als PDF exportierbar sein (1.6).
- *F290* PlagTag muss unterschiedliche Algorithmen für die Plagiatserkennung unterstützen (1.1).
- *F300* Die in der Informatik übliche Zitierweise muss erkannt werden (1.1).
- *F310* Die nicht in der Informatik üblichen Zitierweisen können erkannt werden (1.1).
- *F320* Die Erkennung von Plagiaten erfolgt in einem dreistufigen Prozess (1.1).
- *F321* Bereitstellung mindestens eines Plagiatskandidaten durch den Anwender (1.1).
- *F322* Bereitstellung von Referenzdokumente durch den Anwender (1.1).
- *F323* PlagTag muss eine automatisierte Suche nach Internetquelle ermöglichen (1.1).
- *F324* PlagTag muss eine Plagiatsüberprüfung durchführen (1.1).

- *F325* PlagTag muss automatisch eine Wiederlegung des Plagiatsverdachts durch Auswertung der Referenzdaten ermöglichen (1.1).
- *F330* Zu vergleichende Texte müssen in gleicher Sprache verfasst sein (1.1).
- *F340* Es muss ein linguistisches Verfahren zur Erkennung von Plagiaten für deutsch oder englisch umgesetzt werden (1.1).
- *F350* PlagTag muss einen einsprachigen Vergleich unterstützen (1.1).
- *F351* Es muss deutsch unterstützt werden (1.1).
- *F352* Es muss englisch unterstützt werden (1.1).
- *F360* Das Dateiformat PDF muss unterstützt werden (1.1).
- *F370* Das Dateiformat TXT muss unterstützt werden (1.1).
- *F380* Das Dateiformat HTML sollte unterstützt werden (1.1).
- *F390* Ergebnisse müssen visuell dargestellt werden (1.1).
- *F400* Die visuelle Darstellung muss in Form eines Strichcodes erfolgen (1.1).
- *F410* Es muss eine Einstellungsmöglichkeit für die Fenstergröße existieren (E.2).
- *F420* Es muss eine Beschreibung der Visualisierung, sowie eine Zusammenfassung der Ergebnisse textuell angezeigt werden (E.2).
- *F430* Als Textcodierung muss UTF-8 unterstüzt werden (E.2).
- *F440* Das Literaturverzeichnis muss bei der Referenzsuche berücksichtigt werden (E.1).
- *F450* Das Verfahren zur Erkennung von Plagiaten kann abgebrochen werden (2.4).
- *F460* Es muss einen Expertenmodus geben (2.4).
- *F470* Es muss einen Standardmodus geben (2.4).
- *F480* Die Auswahl von Algorithmen muss im Expertenmodus möglich sein (2.4).
- *F490* PlagTag muss das Hochladen von Plagiatskandidaten unterstützen (2.4).
- *F500* Es muss aus der Visualisierung erkenntlich sein, welche Textpassage plagiiert wurde (2.4).
- *F510* PlagTag muss Apple Safari in der Version 5 unterstützen (2.6).
- *F520* PlagTag muss Google Chrome in der Version 15 unterstützen (2.6).
- *F530* PlagTag muss Mozilla Firefox in Verision 8 unterstützen.
- *F540* Ein Ergebnisdokument muss zu jeder Zeit bearbeitet werden können.
- *F550* Das Design der Universität soll im Webfrontend genutzt werden.
- *F560* Die Internetrecherche muss abgebrochen werden können.
- *F570* PlagTag muss die Möglichkeit bieten aus dem Webfrontend Plagiatsüberprüfungen zu löschen.
- *F580* PlagTag muss die Statushistorie der Plagiatsüberprüfung anzeigen können.

- *F590* Der Anwender kann manuell den Autor der Referenzdaten und der Plagiatskandidaten angeben.
- *F591* Der angegebene Autor wird bei der Internetrecherche von der Suche ausgeschlossen.
- *F600* Die Internetrecherche muss eine Blacklist bei der Internetquellensuche berücksichtigen.
- *F610* Der Anwender kann manuell auswählen, ob es sich bei gefundenen Plagiaten um Plagiate handelt.

## 3.3. Leistungsanforderungen

**Autor:** Tore Bierwirth, Sieglinde Hahn, Maxim Klimenko, Björn Wolff
Die *Leistungsanforderungen* beschreiben in welchem Umfang die Leistungserfüllung erfolgen muss.

- *L010* Als Referenz für die korrekte Funktionsweise der Anwendung wird die Dissertation von zu Guttenberg verwendet (1.1).
- *L020* Die Referenzdokumente für die Überprüfung der zu Guttenberg-Dissertation müssen PlagTag zur Verfügung gestellt werden (1.1).
- *L030* 25% aller im Wiki *GuttenPlag* [Div11] doppelt gesichteten Seiten der Guttenberg-Dissertation müssen erkannt werden (1.1).

## 3.4. Entwurfs- und Schnittstellenanforderungen

**Autor:** Tore Bierwirth, Sieglinde Hahn, Maxim Klimenko, Björn Wolff
Die *Entwurfsanforderungen* stellen die Anforderungen an die Systemarchitektur dar.

- *E010* Komponenten müssen erst hinzugefügt werden, bevor sie verwendet werden können (2.3).
- *E020* Die Websuche muss als Batch durchgeführt werden (E.2).
- *E030* Aus sicherheitsrelevanten Gründen muss die Kommunikation von PlagTag mit dem HERO Cluster über eine verschlüsselte SSH-Verbindung erfolgen (E.4).
- *E031* Ein technischer User ist auf dem HPC-Cluster erforderlich.
- *E040* Es dürfen keine Dienste auf dem Cluster implementiert werden (E.4).
- *E050* Der Grad der Verschleierung muss parametriesierbar sein, um von einem Komplettplagiat differenziert werden zu können (1.5).
- *E060* Die Algorithmen zur Plagiatserkennung müssen parametrisiert sein (1.5).
- *E070* Das Webfrontend muss über Browser aufgerufen und dargestellt werden (2.1).
- *E080* PlagTag muss auf einer Virtuellen Maschine betrieben werden können (2.1).
- *E090* Die Suche nach Referenzdaten muss über HTTP erfolgen (2.1).
- *E100* Das Webfrontend muss über HTTP via TCP/IP erreichbar sein (2.1).
- *E110* Die Benutzeroberfläche muss als Webanwendung realisiert werden (1.1).
- *E120* Die Ausführung der Plagiatserkennung muss, auf Grund ihrer Rechenintensivität, zusätzlich auf dem HERO-Cluster realisiert werden (1.1).

- *E121* Zugriff auf dem Cluster muss aus dem Uni-Netzwerk möglich sein.
- *E130* Objektorientierte Komponenten müssen die Verteilung der Anwendung erleichtern (1.1).
- *E140* Erweiterbarkeit und Austauschfähigkeit einzelner Komponenten muss bestehen (1.1).
- *E150* Das linguistische Verfahren muss generisch sein (1.1).
- *E160* Ein technischer User auf dem MySQL Server muss vorhanden sein.
- *E170* Es muss ein Betriebssystem mit JAVA Unterstützung bereitstehen.
- *E180* Es müssen TCP/IP Kommunikation auf den Ports 80 sowie der Portrange 8000-8800 möglich sein.
- *E190* Es ist ein Server mit Tomcat7 als Dienst erforderlich .

## 3.5. Datenhaltungsanforderungen

**Autor:** Tore Bierwirth, Sieglinde Hahn, Maxim Klimenko, Björn Wolff
Die *Datenhaltungsanforderungen* werden zur Verwaltung der verschiedenen Daten beschrieben.

- *D010* Ergebnisdokumente müssen persistent abgespeichert werden (2.1).

## 3.6. Qualitätsanforderungen

**Autor:** Tore Bierwirth, Sieglinde Hahn, Maxim Klimenko, Björn Wolff
Die *Qualitätsanforderungen* beschreiben die nichtfunktionalen Anforderungen an PlagTag. Dabei werden die externen Qualitätsanforderungen des Anwenders und die internen Qualitätsanforderungen des Entwicklers berücksichtigt.

### 3.6.1. Zuverlässigkeit

Die *Zuverlässigkeit* beschreibt die Genauigkeit der Funktionalität von PlagTag.

- *Q010* Im Fehlerfall müssen Fehlermeldungen ausgegeben werden (2.4).
- *Q020* Korrekt zitierte Zitate dürfen nicht als Plagiate erkannt werden (1.1).

### 3.6.2. Sicherheit

Die *Sicherheit* stellt die Anforderungen an PlagTag, welche aufgrund rechtlicher Aspekte eingehalten werden müssen.

- *Q030* Personenbezogene Daten zum Autor des Plagiatskandidaten müssen anonymisiert werden (2.1).
- *Q040* Referenzdaten müssen nach Beendigung der Überprüfung gelöscht werden (2.4).
- *Q050* Plagiatskandidaten müssen nach der Überprüfung gelöscht werden (2.4).
- *Q060* Der Zugriff auf Dokumente innerhalb von PlagTag muss auf die Person, welche die Dokumente bereitgestellt hat, beschränkt sein (1.1).

Aus den Sicherheitsanforderungen ergibt sich, dass jeweils nur die Person Zugriff auf Dokumente hat, die diese in PlagTag bereitgestellt hat. Deshalb ist eine Zugriffskontrolle notwendig und eine Registrierung der Anwender erforderlich. Hierzu verfügt das Webfrontend über eine Registrierungsmaske, welche die Registrierung und die damit verbundene Speicherung von Anwenderdaten in der

Datenhaltung ermöglicht. Nur registrierte Anwender können sich anmelden und PlagTag nutzen. Zur Registrierung werden folgende Daten des Anwenders benötigt:
**Vor- und Nachname:** Der Vor- und Nachname des Anwenders.
**Benutzername:** Der Anwender kann sich einen Benutzernamen zur Verwendung PlagTags aussuchen.
**Passwort:** Der Anwender kann sich ein beliebiges Passwort aussuchen, um seinen Zugang zu PlagTag abzusichern. Dies wird in Kombination mit dem Benutzernamen bei der Anmeldung abgefragt. Das Passwort wird verschlüsselt in der Datenhaltung gespeichert.
**E-Mail-Adresse:** Die E-Mail-Adresse des Anwenders damit ein neues Passwort generiert und versandt werden kann, sollte der Anwender sein Passwort vergessen. Außerdem kann der Anwender per E-Mail über den Status einer Plagiatsüberprüfung informiert werden.

Diese Daten werden mit den entsprechenden Plagiatsüberprüfungen verknüpft, damit eine eindeutige Zuordnung von Anwendern und Plagiatsüberprüfungen möglich ist. Somit können Plagiatsüberprüfungen nur von den Anwendern ausgeführt und begutachtet werden, die diese erstellt haben.

### 3.6.3. Wartbarkeit

Die *Wartbarkeit* zeigt den Umfang auf, in dem die Evolution von PlagTag erlaubt ist.

- *Q070* Es muss eine parallele Dokumentation zur Entwicklung stattfinden (Quelltext und Projekt) (1.9).
- *Q080* Es muss ein Entwicklerhandbuch geben (2.5).

### 3.6.4. Erlernbarkeit

Die *Erlernbarkeit* hilft, damit neue Anwender sich schnell mit PlagTag zurechtfinden.

- *Q090* Es muss ein Benutzerhandbuch geben (1.6).
- *Q100* Es müssen Usability-Tests durchgeführt werden (2.5).

### 3.6.5. Sonstiges

Die *sonstigen Anforderungen* sollen weitere nichtfunktionale Anforderungen darstellen.

- *Q110* PlagTag muss mindestens einen Plagiatskandidaten zur Verfügung gestellt bekommen (2.2).
- *Q120* Es dürfen keine kostenpflichtigen Dienste verwendet werden (E.2).
- *Q130* PlagTag darf während der Durchführung der PG nicht kommerziell genutzt werden (E.1).
- *Q140* Es dürfen nur Techniken mit ausreichender Dokumentation verwendet werden (1.9).
- *Q150* PDF's dürfen nicht kennwortgeschützt sein (1.1).

## 3.7. Vorgehensanforderungen

**Autor:** Tore Bierwirth, Sieglinde Hahn, Maxim Klimenko, Björn Wolff
Die *Vorgehensanforderungen* beschreiben die Anforderungen, welche die PG im Rahmen der Entwicklung von PlagTag an sich selbst stellt.

- *V010* Anforderungen müssen über die Dokumente hinweg nachvollziehbar sein (1.9).

- *V020* Bei Bedarf müssen Schulungen im Rahmen der PG durchgeführt werden, um die Kompetenzen der einzelnen Teammitglieder anzugleichen (1.9).
- *V030* Das Vorgehen innerhalb des Projektes muss sowohl agil, als auch iterativ geschehen (1.7).
- *V040* Anpassung des Pflichtenhefts und Entwurfs müssen auch rückwirkend erfolgen (1.7).
- *V050* Es sollten drei Zyklen durchlaufen werden (1.7).
- *V060* Ein Zyklus muss folgende Phasen enthalten: Analyse und Design, Implementierung, Test und Reflexion und Fertigstellung (1.7).
- *V070* Für eine effiziente Kommunikation wird ein Ticketsystem verwendet (1.7).
- *V080* Es sollte eine enge Zusammenarbeit mit dem Auftraggeber durch wöchentliche Meetings erfolgen (1.7).
- *V090* Zum Einhalten von Terminen müssen Meilensteine festgelegt werden (1.7).
- *V100* Die Versionsverwaltung muss über Subversion-Repository erfolgen (1.7).
- *V110* Es muss ein Projekthandbuch gepflegt werden (1.7).

# A. Abbildungsverzeichnis


1. Strichcode von Plagaware [Pla12c] . . . 18
2. Status der Überprüfung [Pla12d] . . . 19
3. Strichcode auf GuttenPlag [Div11] . . . 19
4. Vorgehensmodell . . . 20
5. RUP Phasen [Tho08] . . . 24
6. RUP Stakeholder angelehnt an [Kru98] . . . 25
7. Dokumentationsverfahren nach Scheibl angelehnt an [Sch85] . . . 26
8. Stakeholder nach Scheibl angelehnt an [Sch85] . . . 26
9. Blockdiagramm der Anwendung in ihrer Umwelt . . . 31
10. Domänenmodell der Anwendung . . . 32
11. Anwendungsfalldiagramm zur Anwendung . . . 33
12. Verfeinerung des AF1 Plagiatskandidat prüfen . . . 35
13. Plagiatskandidat vorbereiten AF1.1 . . . 36
14. Referenzdaten erheben AF1.2 . . . 37
15. Ergebnisdokument gestalten AF1.4 . . . 39
16. Mockup der Startseite - Standardmodus . . . 40
17. Mockup der Startseite - erweiterter Eingabemodus . . . 41
18. Mockup der Ansicht: Referenzdokumente hochladen . . . 42
19. Mockup der Seite: Ergebnisdokument visualisieren . . . 42
20. Mockup der Seite: Ergebnisdokument visualisieren - Detailansicht des Strichcode . . . 43

# B. Tabellenverzeichnis

1. Übersicht der Anwendungsfälle . . . . . 34
2. Anwendungsfall - Muster . . . . . 35
3. AF1 Plagiatskandidat prüfen . . . . . 36
4. AF1.1 Plagiatskandiadaten vorbereiten . . . . . 37
5. AF1.2 Referenzdaten erheben . . . . . 38
6. AF1.3 Überprüfung durchführen . . . . . 39
7. AF1.4 Ergebnisdokument gestalten . . . . . 40

# C. Literaturverzeichnis


[Bal09] Balzert, Helmut ; Rüdinger, Andreas (Hrsg.) ; Kohl, Kerstin (Hrsg.) ; Fraude, Dagmar (Hrsg.): *Lehrbuch der Softwaretechnik: Basiskonzepte und Requirements Engineering*. Spektrum, 2009

[BBK01] Back, A. ; Becker, J. ; König, W. ; Stürken, M. (Hrsg.): *Lexikon der Wirtschaftsinformatik*. Springer, 2001

[Bee00] Beer, Timo de: *Die Geschichte der EU*. Grin, München, 2000

[Blu03] Blumenwitz, Dieter: *Wer gibt die Verfassung Europas?* Politische Studien, Atwerb, 2003

[Cas01] Casper, Gerhard: *Die Karlsruher Republik*. online. http://www.stanford.edu/group/gcasper_project/cgi-bin/files/papers/karlsruhe.pdf. Version: 2001. – Zugriff am 13.01.2012

[cha11] *FDP-Politiker Chatzimarkakis verliert Doktortitel*. http://www.focus.de/politik/deutschland/wissenschaft-fdp-politiker-chatzimarkakis-verliert-doktortitel_aid_645561.html. Version: 2011. – Letzter Zugriff 22.11.2011

[con12] *ConQAT - Continuous Quality Assessment Toolkit*. online. http://conqat.cs.tum.edu/index.php/ConQAT. Version: 2012. – Zugriff am 19.01.2012

[CSFP] Collins-Sussman, B. ; Fitzpatrick, B. W. ; Pilato, M.: *Version Control with Subversion*. http://svnbook.red-bean.com/. – Letzter Zugriff am 08.08.2012

[Den11] Denkler, Thorsten: *Uni entzieht Koch-Mehrin den Doktortitel*. http://www.sueddeutsche.de/karriere/plagiate-in-der-dissertation-uni-entzieht-koch-mehrin-den-doktortitel-1.1108955. Version: 2011. – Letzter Zugriff 22.11.2011

[Div11] Diverse: *GuttenPlag Blog*. online. http://de.guttenplag.wikia.com/wiki/GuttenPlag_Wiki. Version: 2011

[EK] Elektronik-Kompendium: *SSH - Secure Shell*. http://www.elektronik-kompendium.de/sites/net/0906061.htm. – Letzter Zugriff am 08.08.2012

[eph12] *Ephorus*. online. https://www.ephorus.com/en/try-now/visuals. Version: 2012. – Zugriff am 19.01.2012

[FHP02] Fuchs, Michael ; Hartleif, Sylvia ; Popovic, Vesna: *Der Weg zum EU-Verfassungskonvent. Berichte und Dokumentationen*. Deutscher Bundestag, Referat Öffentlichkeitsarbeit, Berlin, 2002

[Fre03] Frey, Bruno: Publishing as Prostitution? – Choosing Between One's Own Ideas and Academic Success. In: *Public Choice* 116 (2003), Nr. 1-2, 205-223. http://www.springerlink.com/content/qg0437183m430225/fulltext.pdf

[Fre05] Frey, Bruno: Problems with Publishing: Existing State and Solutions. In: *European Journal of Law and Economics* 19 (2005), Nr. 2, 173-190. http://www.springerlink.com/content/r24503g06v4x5791/fulltext.pdf

[fre12] *Alleged self plagiarism by Bruno Frey et al.* online. https://docs.google.com/spreadsheet/pub?hl=de&key=0AuEtgCUuVBDUdFlVc3ZIR2dsRFd1d29iWndjNVdVSFE&hl=de&gid=1. Version: 2012. – Zugriff am 29.01.2012

[GB09] Grechenig, Thomas ; Bernhart, Mario: *Softwaretechnik: Mit Fallbeispielen aus realen Entwicklungsprojekten*. Pearson, 2009

[GDM09] Gunda Dreyer, Jost K. ; Meckel, Astrid: *Urheberrecht: Urheberrechtsgesetz, Urheberrechtswahrnehmungsgesetz, Kunsturhebergesetz (Heidelberger Kommentar)*. C.F. Müller, 2009

[Gut09] Guttenberg, Karl-Theodor zu: *Verfassung und Verfassungsvertrag. Konstitutionelle Entwicklungsstufen in den USA und der EU*. Duncker & Humblot, Berlin, 2009

[gut11] *PlagiatsKategorien*. online. http://de.guttenplag.wikia.com/wiki/PlagiatsKategorien. Version: 2011. – Zugriff am 25.10.2011

[gut12a] *Fragment 071 27-29*. online. http://de.guttenplag.wikia.com/wiki/Fragment_071_27-29. Version: 2012. – Zugriff am 12.01.2012

[gut12b] *Guttenberg-2006/048*. online. http://de.guttenplag.wikia.com/wiki/Guttenberg-2006/048. Version: 2012. – Zugriff am 12.01.2012

[gut12c] *Guttenberg-2006/071*. online. http://de.guttenplag.wikia.com/wiki/Guttenberg-2006/071. Version: 2012. – Zugriff am 12.01.2012

[gut12d] *Guttenberg-2006/077*. online. http://de.guttenplag.wikia.com/wiki/Guttenberg-2006/077. Version: 2012. – Zugriff am 12.01.2012

[gut12e] *Guttenberg-2006/149*. online. http://de.guttenplag.wikia.com/wiki/Guttenberg-2006/149. Version: 2012. – Zugriff am 12.01.2012

[gut12f] *Guttenberg-2006/205*. online. http://de.guttenplag.wikia.com/wiki/Guttenberg-2006/205. Version: 2012. – Zugriff am 29.01.2012

[gut12g] *Guttenberg-2006/330*. online. http://de.guttenplag.wikia.com/wiki/Guttenberg-2006/330. Version: 2012. – Zugriff am 12.01.2012

[gut12h] *Seite 008-009 Gliederung*. online. http://de.guttenplag.wikia.com/wiki/Seite_008-009_Gliederung. Version: 2012. – Zugriff am 12.01.2012

[Hä06] Häberle, Peter: *Europäische Verfassungslehre*. Helbing und Lichtenhahn, Baden-Baden, Basel, 2006

[IA] ITAdministrator: *Softwareverteilung*. online. http://www.it-administrator.de/themen/server_client/grundlagen/110904.html

[IBM02] IBM: *Application Maintenance Turnover Package*. 2002

[Ken62] Kennedy, John F.: *The Goal of an Atlantic Partnership*. Rede in Philadelphia am 4. Juli 1962, 1962

[Kla11] Klaube, Jürgen: *Vgl. auch Guttenberg 2009*. http://www.faz.net/aktuell/feuilleton/plagiats-affaere-vgl-auch-guttenberg-2009-1597256.html. Version: 2011. – Letzter Zugriff 22.11.2011

[Kru98] Kruchten, Philippe: *The Rational Unified Process an Introduction*. Addison-Wesley Longman, 1998

[KS98] Kotonya, Gerald ; Sommerville, Ian: *Requirements Engineering, Processes and Techniques*. 1998

[KWW10] Köhler, Katrin ; Weber-Wulff, Debora: Plagiatserkennungstest 2010 / HTW Berlin. 2010. – Forschungsbericht

[MD08] Mens, Tom ; Demeyer, Serge: *Software Evolution*. Springer Berlin Heidelberg, 2008

[OAS] OASIS: *OASIS Service Component Architecture / Assembly*. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=sca-assembly. – Zugriff am 23.04.2012

[pla12a] *PlagiarismFinder*. online. http://www.plagiarismfinder.de/produkte/versionen. Version: 2012. – Zugriff am 19.01.2012

[pla12b] *PlagiatCheck.de - Sofortige, automatische & kostenlose Analyse*. online. http://www.plagscan.com/plagiatcheck/. Version: 2012. – Zugriff am 19.01.2012

[Pla12c] Plagaware: *Plagiatsprüfung und Quellennachweis von Artikeln, Manuskripten und Hausarbeiten*. online. http://www.plagaware.de/informationen/einsatzgebiete/plagiatspruefung-hausarbeiten. Version: 2012. – Letzter Zugriff 26.01.2012

[Pla12d] Plagiarisma: *Plagiarisma.Net*. http://plagiarisma.net/. Version: 2012. – Zugriff am 19.01.2012

[Pla12e] PlagScan: *PlagScan*. online. http://www.plagscan.com/. Version: 2012. – Zugriff am 27.01.2012

[RCK09] Roy, Chanchal K. ; Cordy, James R. ; Koschke, Rainer: *Comparison and Evaluation of Code Clone Detection Techniques and Tools: A Qualitative Approach*. online. http://www.cs.usask.ca/~croy/papers/2009/RCK_SCP_Clones.pdf. Version: 2009. – Zugriff am 13.01.2012

[Sch85] Scheibl, Hans-Jürgen: *Wie dokumentiere ich ein DV-Projekt? - Dokumentationsverfahren in Theorie und Praxis*. Expert Verlag, 1985

[Sch08] Schwartmann, Rolf: *Praxishandbuch Medien-, IT- und Urheberrecht*. C.F. Müller, 2008

[Sch11] Schimmel, Roland: *Von der hohen Kunst ein Plagiat zu fertigen - Geleitwort Karl Theodor zu Guttenberg*. Lit Verlag, 2011

[Sne04] Sneed, Harry: *Software-Produktmanagement: Wartung und Weiterentwicklung bestehender Anwendungssysteme*. Dpunkt Verlag, 2004

[Tec] TechTerms: *PDF*. http://www.techterms.com/definition/pdf. – Letzter Zugriff am 08.08.2012

[Tho08] Thorsten Horn: *RUP*. online. http://www.torsten-horn.de/img/RUP.jpg. Version: 2008

[tur12] *Turnitin*. online. https://turnitin.com/static/videos/demo_tiisuite.html. Version: 2012. – Zugriff am 19.01.2012

[urk12] *Urkund Beta*. online. https://secure.urkund.com/beta/view/4210713-632459-994257. Version: 2012. – Zugriff am 19.01.2012

[VS01] Volkmann-Schluck, Sonja: *Die Debatte um eine europäische Verfassung*. CAP-Working Paper, München, 2001

[Was97] Wasser, Hartmut: *Amerikanische Präsidialdemokratie*. Franzis, München, 1997

[Wee05] Weege, Wilhelm: *Der Weg zum EU-Verfassungskonvent. Berichte und Dokumentationen.* Deutscher Bundestag,Wissenschaftliche Dienste, Berlin, 2005

[WHKL08] Wütherich, Gerd ; Hartmann, Nils ; Kolb, Bernd ; Lübken, Matthias: *Die OSGi Service Platform-Eine Einführung mit Eclipse Equinox.* dpunkt Verlag, 2008

# D. Glossar

**Alibi-Fußnote** Eine Alibi-Fußnote liegt vor, wenn längere Textbereiche übernommen, aber nur ein Teilbereich korrekt referenziert wird [gut11].

**Dashboard** Dashboard ist die englische Übersetzung für Armaturenbrett und ist eine weit verbreitete Technik um komprimiert eine Menge von Informationen visuell darzustellen. Auf einem Dashboard sind einzelene Bereiche (Miniprogramme, Visualisierungen) plaziert, die selbst interaktiv arbeiten können.

**Datenhaltung** Die Datenhaltung dient für die Anbindung an eine Datenbank, wobei noch nicht entschieden wurde, welche Datenbank verwendet werden soll.

**Deployment** Deployment oder Softwareverteilung genannt, umfasst alle Aktivitäten, welche bei der Übergabe der Software benötigt werden [IA].

**Doppelschöpfung** Eine Doppelschöpfung liegt vor, wenn von zwei Autoren vollkommen unabhängig voneinander eine identische Leistung geschaffen wird [gut11].

**DSB** DSB steht für Datenschutzbeauftragter und stellt eine Institution in einer öffentlichen Institution oder einem Privatunternehmem dar, welche sich um alle Belange des Datenschutzes kümmert. Diese Institution vertritt die Universität nach außen, berät die Fachabteilungen und überwacht die Datenschutzrechte der Studierenden und Mitarbeiter gegenüber der Hochschule.

**Eigenplagiat** Ein Eigenplagiat liegt vor, wenn ein Autor eigene Quellen verwendet, ohne diese entsprechend zu zitieren [gut11].

**Ergebnisdokument** Das Ergebnisdokument speichert als Prüfbericht die Informationen aus der Plagiatsüberprüfung eines Plagiatskandidaten. Zu den Informationen gehören: Anzahl der Plagiate, wo wurden die Plagiate gefunden und welche Referenzdaten bzw. Internetquellen wurden plagiiert.

**Erkennungs-Algorithmen** Die Erkennungs-Algorithmen sind Algorithmen zur Erkennung von Plagiate. Diese sind derzeit noch nicht bestimmt.

**Fair-Share Prinzip** Das Fair-Share Prinzip ist Bestandteil der Software-Lösung des HPC-Cluster, welche sich um die Aufgabenplanung einzelner Programme auf dem Cluster kümmert.
Hierauf bezogen, bezeichnet Fair-Share die gerechte Aufteilung der Rechenleistung zwischen dein einzelnen Fakultäten der Universität. Diese ist je nach Einstellung zwischen den drei Fakultäten gedrittelt. Dies bedeutet, jede Institution hat ein Anrecht auf 33% der Rechenleistung, betrachtet über einen festgelegten Zeitraum.

**False Positive** Ein False Positive ist in dieser Anwendung ein erkanntes Plagiat, welches aber in Wirklichkeit keins ist, da es korrekt zitiert wurde.

**Halbsatzflickerei** Eine Halbsatzflickerei liegt vor, wenn einzelne Wörter oder Satzfragmente ohne korrekte Quellenangabe übernommen werden.

**HPC-Cluster** Ein High Performance Computing Cluster (HPC Cluster oder vereinfacht *Cluster*) stellt ein Synonym für einen Höchstleistungsrechner dar, der verteilt komplexe Aufgaben bewältigt.

**HTML** Hyper Text Markup Language ist eine Web-Auszeichnungssprache, ist Grundlage des World Wide Web und wird von einem Webbrowser dargestellt. Aktuell befindet sich HTML Version 5 in der Standardisierung.

**Internetquelle** Die Internetquellen werden durch eine automatische Internetrecherche durch PlagTag gefunden. Dabei soll PlagTag Webseiten finden, die durch den Ersteller des Plagiatskandidat plagiiert wurden.

**Internetrecherche** Die Internetrecherche ist eine Komponente der Anwendung PlagTag. Diese dient dem Finden von Referenzdaten im Internet.

**Komplettplagiat** Ein Komplettplagiat liegt bei einer nahezu wörtlichen Übereinstimmung des Plagiatskandidaten zu einem Referenzdokument ohne korrekte Quellenangabe vor [gut11].

**Komponente** Eine Komponente (hier: Software-Komponente) ist ein Software-Element, das konform zu einem Komponentenmodell (z.B. aus der UML) ist und über Schnittstellen mit anderen Komponenten verknüpft und ausgeführt werden kann. Komponentenbasierte Software wird z.B. mittels Web-Services, CORBA, Enterprise Java Beans (EJBs) oder COM entwickelt.

**Kopiertes Zitat** Ein kopiertes Zitat liegt vor, wenn von einem bereits zitierten Text ein weiteres Zitat angefertigt wird und dabei nicht der Urheber, sondern nur ein Zitierende, referenziert wird [gut11].

**Mockup** Ein Mockup ist ein Modell des Produkts, hier der Grafischen Benutzeroberfläche.

**OSGi** OSGi Alliance ist eine hardwareunabhängige dynamische Softwareplattform, die es erleichtert, Anwendungen und ihre Dienste per Komponentenmodell (*Bundle*/*Service*) zu modularisieren und zu verwalten (*Service Registry*). Die OSGi-Plattform setzt eine Java Virtual Machine (JVM) voraus und bietet darauf aufbauend das OSGi-Programmiergerüst [WHKL08].

**Parodie** Eine Parodie liegt vor, wenn basierend auf einer vorhandenen textuellen Struktur eine vollständig neue Leistung erbracht wird [gut11].

**PDF** Das Portable Document Format (PDF) ist ein plattformunabhängiges Dateiformat für Dokumente, das vom Unternehmen Adobe Systems entwickelt und 1993 veröffentlicht wurde. Ziel ist die originalgetreue Wiedergabe auf allen Plattformen [Tec].

**Plagiatserkennung** Die Plagiatserkennung meint das Ausführen der Erkennungs-Algorithmen auf einen Computer, um Plagiate zu identifizieren.

**PlagTag** PlagTag ist der Name der Plagiatserkennungssoftware.

**Plagiat** Ein Plagiat liegt dann vor, wenn jemand vorgibt, ein Werk selbst erstellt zu haben, obwohl die Inhalte ganz oder teilweise aus fremden Quellen stammen, die falsch oder unvollständig zitiert wurden. Je nachdem wie die Zitiertechnik falsch verwendet wurde, lassen sich die Plagiate zusätzlich genauer kategorisieren Je nachdem wie die Zitiertechnik falsch verwendet wurde, lassen sich die Plagiate zusätzlich genauer kategorisieren. Es existieren vielfältige Formen geistiger Leistung, z.B. Texte, Bilder oder Videos.

**Plagiatsfund** Unter einem Plagiatsfund wird das Entdecken eines oder mehrere Plagiat verstanden.

**Plagiatskandidat** Ein Plagiatskandidat ist ein Dokument, welches vom Anwender in das System eingefügt und dort auf Plagiate überprüft wird.

**Plagiatsüberprüfung** Die Plagiatsüberprüfung beschreibt das Vorgehen in PlagTag, welches den gesamten Ablauf der Anwendung ab dem Anlegen einer Überprüfung vom Anwender bis zur Ausgabe des Ergebnisses der Überprüfung meint.

**Referenzdaten** Die Referenzdaten sind der Sammelbegriff für Referenzdokumente und Internetquellen.

**Referenzdokument** Die Referenzdokumente sind die Dokumente, welche durch den Anwender zusätzlich neben dem Plagiatskandidat in das System eingefügt werden, um genau diese Dokumente mit dem Plagiatskandidat zu vergleichen.

**SCA** Die Service Component Architecture (SCA) ist eine Sammlung an Spezifikationen, welche ein Modell einer Serviceorientierten Architektur (SOA) beschreiben. SCA basiert auf offenen Standards wie Web Services. SCA-Komponenten sind unabhängig von einer konkreten Technologie. [OAS].

**Server-System** Wird der Begriff des Server-Systems ohne explizite Einschränkungen verwendet, so bezieht sich dieser immer auf die verfügbare Architektur.

**Shake and Paste** Ein Shake and Paste liegt vor, wenn einzelne Sätze oder Absätze, ohne korrekte Quellenangabe, übernommen werden [gut11].

**SSH** SSH ist die Abkürzung für Secure Shell. Es bezeichnet ein Netzwerkprotokoll sowie verschiedene Programme, mit denen man eine verschlüsselte Netzwerkverbindung zwischen mehreren Rechnern aufbauen kann. Dieses Protokoll kommt primär in der Administration von UNIX-basierten Computern zum Einsatz [EK].

**Strichcode** Mithilfe eines Strichcodes werden die Menge und die Position der Plagiate im Text deutlich. Dabei werden verschiedene Farben verwendet, wobei jede Farbe für eine andere plagiierte Textpassage steht, somit wird auch die Länge der Plagiat anhand des Strichcodes deutlich.

**Strukturplagiat** Ein Strukturplagiat liegt vor, wenn z.B. Auflistungen oder Bereiche des Inhaltsverzeichnisses übernommen werden und somit auch die wiedererkennbare Struktur, ohne korrekte Quellenangabe [gut11].

**Subversion-Repository** Subversion ist eine Software zur Versionsverwaltung von Dokumenten und Programmen [CSFP]. Repository beschreibt das Verzeichnis, in dem Dokumente und Programme mit zusätzlichen Metadaten gespeichert werden, um eine Versionverwaltung zu ermöglichen [BBK01].

**System** Wird der System-Begriff ohne explizite Einschränkungen verwendet, so bezieht sich dieser immer auf das im Projekt zu entwickelnde Gesamtprodukt. Dieses teilt sich auf in eine zentrale Berechnungs-Komponente und ein angebundenes Webfrontend.

**Trac** Ein webbasiertes Projektmanagement-Werkzeug. Beinhaltet ein Wiki zur Dokumentenverwaltung.

**TXT** TXT bezeichnet reinen Fließtext ohne Formatierung.

**unbewusste Entlehnung** Eine unbewusste Entlehnung liegt vor, wenn ein Autor einen Text gelesen hat und diesen nach einiger Zeit subjektiv als den Eigenen empfindet [gut11].

**Usability-Test** Ein Usability-Test prüft die Benutzungstauglichkeit eines Softwaresystems mit potenziellen Anwendern. Diese erhalten i.d.R. Testfälle, bei deren Bearbeitung diese beobachtet und mögliche Probleme aufgedeckt werden [GB09].

**Verschleierung** Eine Verschleierung liegt bei einem umformulierten Text ohne korrekte Quellenangabe vor [gut11].

**vertikaler Durchstich** Der vertikale Durchstich wird dazu verwendet ausgewählte Funktion des Produktes zu zeigen. Somit können bestimmte Anwendungen getestet werden. Der Einsatz findet in einer frühen Phase des Designs statt.

**Webfrontend** Das Webfrontend stellt ein System für den Anwender auf der Clientseite visuell dar. Es repräsentiert die Benutzeroberfläche (GUI).

**Wiki** Eine Plattform, welche die Verwaltung von Dokumenten über das Internet ermöglicht.

# E. Anhang

## E.1. Interview mit Jan Jelschen

Das erste Interview fand am 25. November 2011 bei Jan Jelschen im Büro statt. Anwesende waren Jan Jelschen (Interviewter als Anwender und Entwickler), Christoph Gerken (Interviewleitung) und Björn Wolff (Protokoll).

Jan Jelschen hat eine Landesstelle als wissenschaftliche Mitarbeiter mit Lehrverpflichtung an der Universität Oldenburg inne. Diese umfasst die Betreuung von Tutorien, sowie die Leitung des Übungsbetriebs zu Vorlesungen. Außerdem werden Bachelor und Diplomarbeiten betreut. Zusätzlich bereitet sich Jan Jelschen auf seine Dissertation vor.
Der Interviewpartner identifizierte als Rollen die Professoren, wobei er hauptsächlich mit Prof. Dr. Andreas Winter arbeitet. Weitere Rollen stellen das Sekretariat der entsprechenden Abteilung, die Studierenden, die Hausverwaltung(Raumbüro/Hausmeister), der Webmaster der Informatikseite, die Bibliotheksdienste und die Personalabteilung dar, welche allerdings nicht mit der zu entwickelnden Anwendung arbeiten werden.
Es wurde geschildert, dass viele Plagiatsfälle in letzter Zeit aufgekommen sind, dabei wurden folgende Vorfälle speziell beschrieben:

- In Seminaren gab es drei Fälle in denen Plagiate aufgetreten sind, was als Motivation für die Projektgruppe gesehen wurde.
- In Zwischenabgaben von Bachelorarbeiten, sowie in Proposals zu Bachelorarbeiten und Dissertationen, wurden Plagiate entdeckt.
- In dem Modul Softwaretechnik II wurden vermehrt Plagiate bei einem Studenten entdeckt.

Diese Plagiate waren zumeist *schlecht* plagiiert, das heißt es wurde wörtlich abgeschrieben, beziehungsweise wurden komplette Absätze ohne Veränderung kopiert.

Jan Jelschen nannte Ephorus als eine Möglichkeit zur automatischen Plagiatsprüfung, wobei aber auch die manuelle Suche mit Google durchgeführt wird.
Die Dateien werden Ephorus im StudIP als PDF zur Verfügung gestellt, wobei auch andere Formate möglich seien. Dann werden die Dateien von Ephorus geprüft und mögliche Plagiate ausgegeben. Außerdem sei eine Ordnerüberwachung durch Ephorus im StudIP möglich.
Als Kritikpunkt wurde die Datenauswahl genannt, die bei Ephorus auf HTML-Seiten begrenzt ist. Zusätzlich ist es möglich, dass bei einer manuellen Googlesuchanfrage sogar noch weitere Quellen gefunden werden, die Ephorus nicht gefunden hat.
Ein weiterer Kritikpunkt war die Erkennungsgenauigkeit von Ephorus. Die Anzahl der von Ephorus erkannten Plagiate seien gering und meist offensichtliche Plagiate. Außerdem werden viele False Positives, also Textstellen, die fälschlicherweise als Plagiate erkannt wurden, geliefert. An dieser Stelle erwähnte Jan Jelschen auch, dass ihm die Erkennungsgenauigkeit wichtiger sei als die Laufzeit. Zusätzlich ist die Laufzeit von Ephorus mit 24 Stunden für ein Dokument sehr lang. Die Prüfung der Guttenberg-Dissertation benötigte sogar weit über 4 Tage (im Nachhinein fast 2 Wochen).
Die Visualisierung der Ergebnisse seien bei Ephorus vor allem zu unübersichtlich. Zwar gäbe es Text zum Durchscrollen, aber eine Gegenüberstellung zwischen Plagiatskandidat und Quelle würde fehlen. Insgesamt würden Visualisierungsmethoden fehlen, wie beispielsweise Barcodes oder Texte, die sich Anklicken lassen.
Die Benutzerfreundlichkeit von Ephorus war für die Funktionalität in Ordnung.

Eine Übersicht über das zu prüfende Dokument mit möglicherweise Anwahl von Textpassagen wurde als Wunsch für eine neue Software genannt. Dabei soll es möglich sein, eine Gegenüberstellung zwischen Plagiat und Dokument vorzunehmen. Besonderes Augenmerk liegt auf der Erweiterbarkeit

der neuen Software, da Teile dieser möglicherweise für die eigene Dissertation genutzt werden sollen.

Der Interviewte arbeitet hauptsächlich mit Zitaten, wie sie in der Informatik verwendet werden, das heißt Tags mit Bezeichnern oder Zahlen (z.B. [AW01], [1]). Dabei sind die eigentlichen Quellenangaben am Ende der Arbeiten unter den Tags zu finden. Die Zitate werden meist an ganze Absätze oder Abschnitte angegeben. Die wörtlichen Zitate, beispielsweise bei kompletten Absätzen, kommen oft bei Studenten vor, aber sind in eigentlichen wissenschaftlichen Arbeiten nicht üblich.
Meist seien Plagiate an der Struktur der Texte zu erkennen, zum Beispiel an der Kommasetzung. Es wird zwanghaft versucht Sätze durch Umstellen zu verändern, was sehr auffällig sei. Die plagiierten Sätze stechen hervor, da sie in vielen Fällen besser formuliert sind als der Rest der Ausarbeitung. Auffällig sind außerdem Veränderungen im Vokabular, diese können dazu führen, dass genauer nachgeschaut wird.
Die meisten Plagiate lassen sich schon dadurch finden, dass Sätze bei Google eingegeben werden. Die Quellen, von denen plagiiert wird, seien in den meisten Fällen nur Webseiten und keine wissenschaftlichen Paper oder ähnliches. Die Studierenden, die plagiieren, würden sich nicht die Mühe machen und wissenschaftliche Arbeiten lesen, sondern wollen die Aufgaben schnell bearbeiten. Hierbei werden auch selten PDFs heruntergeladen und plagiiert. Als weiteren Anhaltspunkt wurde die Überprüfung von Referenzliteratur genannt, die bei Seminaren angegeben wird. Allgemein würde aber zu wenig gegen Literatur geprüft werden, da dies ein zu großer zeitlicher Aufwand wäre. Somit wäre es auf jeden Fall ratsam das Literaturverzeichnis durchzugehen.

Als Entwicklungssprache würde Jan Jelschen Java favorisieren, als Umgebung kämen daher NetBeans oder Eclipse in Frage. Als Framework zur Erstellung der komponentenbasierten Software könnten das Open Services Gateway initiative Framework (OSGi) oder die Service Component Architecture (SCA) genutzt werden.

Als mögliche Vergleichsdaten wurden folgende Möglichkeiten erwähnt:

- Webseiten,
- Referenzdokumente,
- Literaturverzeichnis,
- Quellen, die in der Bibliothek oder in der Onlinebibliothek der Universität (IEEE, Springer) verfügbar sind.

Der Studierende hat das Urheberrecht für seine Ausarbeitung, daher ist auch Ephorus, im Bezug auf Datenschutz, schon fragwürdig. Des Weiteren sei es problematisch, Bachelorarbeiten woanders als auf dem eigenen Rechner zur Verfügung zu stellen. Der Interviewte sieht Webquellen als eher unproblematisch an. Es sollte keine kommerzielle Nutzung der Software außerhalb der Veranstaltung geben. Falls *illegale* Quellen bei der Websuche gefunden werden, soll der Anwender eine Entscheidung treffen. Außerdem wäre es möglich eine Blacklist zu erstellen, oder einen Disclaimer zu schreiben. Jedoch ist es Jan Jelschen nicht bekannt, inwieweit solche Quellen urheberrechtlich problematisch werden könnten. Am besten soll nur der Report gespeichert und die Referenzen nach Suche gelöscht werden.

## E.2. Interview mit Andreas Winter

Das zweite Interview fand am 28. November 2011 bei Andreas Winter im Büro (A2 2-220) statt. Anwesende waren Andreas Winter (Interviewter als Anwender), Marion Gottschalk (Interviewleitung) und Sieglinde Hahn (Protokoll).

Andreas Winter ist Professor an der Universität Oldenburg und betreut die PG *Clonebusters*, daher steht er mit der Rolle des Professors direkt in Verbindung mit der zu erstellenden Anwendung. Weitere Rollen, die in dem Interview identifiziert wurden und mit der Anwendung in Beziehung stehen, sind WiMis, Tutoren, die Fachschaft und der Datenschutzbeauftragte. Aus eigener Erfahrung konnte Andreas Winter berichten, dass in einigen Seminar- und Abschlussarbeiten Plagiatsfälle aufgetreten sind. In einem Proposal für eine Promotion wurde sogar ein Paper von ihm selbst plagiiert.

Ein Plagiatsverdacht ergibt sich meist durch einen Stilbruch im Text, woraufhin bei Google recherchiert wird, um die entsprechende Textpassage zu finden. Das bisherige System *Ephorus* konnte bei der Plagiatssuche bisher nicht als sehr verlässlich eingeordnet werden. Die Funktionsweise von Ephorus sieht wie folgt aus:

1. Eine Datei muss ins Stud.IP hochgeladen werden.
2. Daraufhin muss das Plug-In Ephorus ausgewählt und die Überprüfung für bestimmte Dateien hier gestartet werden.
3. Nach einem längeren Warteprozess, welcher mehrere Tage andauern kann, wird eine Prozentauswertung angezeigt.
4. Daraufhin kann eine Gesamtanzeige oder eine Einzelanzeige ausgewählt werden. Plagiierte Textpassagen werden rot markiert.

Ein genannter Nachteil an Ephorus ist das Erkennen des Literaturverzeichnisses als Plagiat. Ein weiterer Nachteil ist die unübersichtliche Darstellung der Plagiate im Stud.IP aufgrund der nicht anpassbaren Textfeldern, in denen der Plagiatskandidat und die Referenzendaten angezeigt werden.

Andreas Winter sieht die Datenbasis für Referenzen, welche mit dem Plagiatskandidaten verglichen werden, in Büchern und Zeitschriften in elektronischer Form. Lehrbücher werden eher selten zur Hand von den Studenten genommen. Die Visualisierung der Anwendung stellt sich Andreas Winter ähnlich wie dem Barcode bei GuttenPlag vor.

Neben dem Wunsch einer *anständigen* Visualisierung soll die Anwendung drei Phasen der Plagiatserkennung durchlaufen:

1. Quellen zum Vergleich, zum Recherchieren und Testen werden angezeigt und weitere Quellen können optional hinzugefügt und bestimmte Quellen zum Vergleich ausgewählt werden.
2. Mustererkennungsverfahren, Bewertung von Verfahren, welche erkennen, ob es ein Plagiat ist oder nicht.
3. Entscheidung, ob es sich um ein Plagiat handelt, welche eingeschränkt automatisiert stattfinden soll, da viele Zitiertechniken existieren.

Außerdem muss ein Handbuch zur neuen Anwendung erstellt werden, um dem Anwender die Bedienung der Anwendung zu erleichtern, besonders bei nicht automatisierten Prozessen.

Nach der Meinung von Andreas Winter werden die Zitiertechniken *inline* und *Fußnote* unterschieden. In der Informatik wird hauptsächlich *inline* verwendet und ist daher für die neue Anwendung wichtiger. Die *Fußnote* wird eher bei den Geisteswissenschaften verwendet.

Da die Anwendung am Ende über ein Webfrontend zu bedienen ist, muss laut Andreas Winter der Browser *Safari* unterstützt werden. Zum Plagiatsvergleich sollen die Datenformate PDF und HTML berücksichtigt werden.

Selbst ist Andreas Winter bzgl. der Problematik mit der Datenverarbeitung/-speicherung bekannt, dass viele Papers im Internet zur Verfügung stehen, obwohl das Copyright bei einem Verlag liegt, da die Autoren selbst ihre Paper auf Webseiten einstellen. Daher ist ihm unklar, ob die Speicherung solcher *frei verfügbaren* Paper erlaubt ist.

Abschließend wurden noch folgende Wünsche zur neuen Anwendung gestellt:

- Plagiate sollen gefiltert werden (wenige False Positives)
- eine farbliche Visualisierung der Plagiate
- Einstellen der Fenstergrößen soll möglich sein
- zwei verschiedene Versionen der Anwendung (Standard- und Expertenmodus)
- Der Expertenmodus soll z.B. die Möglichkeit bieten Algorithmen und verschiedene Parameter bestimmen zu können.

## E.3. Interview mit dem Datenschutzbeauftragten

Das dritte Interview fand am 28. November 2011 mit Thomas Geuken in Raum V01-0-010 statt. Geführt wurde dieses von Tore Bierwirth (Protokoll) und Christian Wübbeling (Interviewer).
Thomas Geuken ist Jurist, er tritt als Anforderungsgeber *Datenschutzbeauftragter* auf und soll helfen, mögliche rechtliche Konsequenzen aus dem Einsatz der Anwendung einschätzen zu können. Aus dem Gespräch ergaben sich folgende Feststellungen:

Dritt-Quellen (z.B. Fachliteratur) dürfen innerhalb des Systems als Referenzdaten im äußersten Fall nur anlassbezogen und temporär gespeichert werden. Sie müssen nach Verwendung gelöscht werden. Ausgenommen sind gemeinfreie Werke wie Gesetzestexte oder Werke, deren Urheber seit länger als 60 Jahren verstorben ist.
Die Verwendung von studentischen Arbeiten als Referenzdaten ist ohne explizite Zustimmung der Studierenden ausgeschlossen.
Studentische Arbeiten dürfen jedoch auf Plagiate hin auch elektronisch überprüft werden, da der/-die StudentIn implizit mit der Abgabe seine/ihre Zustimmung erteilt.
Prüf-Protokolle enthalten zwar ebenfalls teilweise geistiges Eigentum, hier handelt es sich jedoch um eine Zulässige Wiedergabe in Auszügen. Die Protokolle dürfen somit analog zur Aufbewahrungszeit der Prüfungsunterlagen gespeichert werden.
Such-Anfragen innerhalb der Software, bei denen Texte an externe Anbieter (z.B. Google) gesendet werden, sind zu anonymisieren, sodass keine personenbezogenen Daten übertragen werden.
Innerhalb der Hochschule dürfen nur dienstrechtlich dazu befugte Mitarbeiter Einsicht in Ergebnisdokumente (Prüfberichte) nehmen. Eine Einsichtnahme in Arbeiten fremder Fachgebiete ist daher fragwürdig. Der Datenschutzbeauftragte empfiehlt, keine Bibliotheksfunktion vorzusehen und die Einsicht auf Erst- und Zweiprüfer zu beschränken.
Ein Zusammenschluss mehrerer Hochschulen und die Verwendung eines Pools von studentischen Referenzdaten ist möglich, bedarf aber ebenso einer expliziten Zustimmung der betroffenen studierenden.

Aufgrund von Unklarheiten stellte der Datenschutzbeauftragte in seiner E-Mail vom 05. Dezember 2011 noch folgendes klar:
Eine Delegation von Zuständigkeiten, hier der Einsichtnahme in Ergebnisdokumente, ist ohne explizite Zustimmung des/der Studierenden nicht zulässig.

## E.4. Interview mit Reinhard Leidl

Das vierte Interview wurde am 12.01.2012 mit Herrn Leidl, dem Clusterbeauftragten der Universität, welcher Mitarbeiter der Abteilung *Wissenschaftliches Rechnen* ist, im PG-Raum (A2 2-219) geführt. Anwesende waren neben dem Interviewten, Christoph (Interviewleitung) und Sieglinde Hahn (Protokoll). In diesem Interview wurden das Cluster und die Funktionsweise näher erläutert. Zu Beginn wurde das Ziel der Projektgruppe beschrieben und das daraus resultierende Interesse an der Clusternutzung.
Herr Leidl hat über den Einsatz des Clusters berichtet, genutzt werden kann der Cluster von den Fakultäten zwei und fünf. Verwendete Programmiersprachen sind C, C++ und Fortran, zusätzlich ist Runtime installiert was die Nutzung von Java möglich macht. Der Cluster kann nicht für die Programmentwicklung genutzt werden, nur für lauffähige Programme. Die Programmentwicklung muss auf lokalen Systemen stattfinden. Die einzelnen Jobs werden in eine Warteschleife gelegt und nacheinander, je nach gewünschter Leistung (Prozessoren, Arbeitsspeicher, Rechendauer), abgearbeitet. Die MPI-Umgebung bekommt mitgeteilt welche Knoten zur Verfügung stehen. Die Auslastungsskripte des Systems können sich mithilfe des passenden Kommandos angesehen werden. Die Ressourcen können von jedem genutzt werden, die Verteilung findet nach dem Fair-Share Prinzip statt. Die Bearbeitung der Jobs ist am effizientesten, wenn Berechnungen parallelisiert stattfinden, daher ist es sinnvoll die zu überprüfenden Plagiatstexte aufzuteilen. Es handelt sich beim Cluster um ein privates Subnetz, welches nicht von außen zugänglich ist. Die Kommunikation zu externen System kann ausschließlich über den SSH-Port durchgeführt werden. Das Bereitstellen eines Dienstes über den die Kommunikation abgewickelt werden könnte ist nicht möglich.

## E.5. Interview mit Andreas Winter 2

Am 16. und 20. Januar 2012 fanden zwei weitere Kurzinterviews mit Andreas Winter statt. Anwesend war am 16. Januar die gesamte Gruppe, am 20. Januar Christian Wübbeling.

Marion Gottschalk stellte in dem Termin am 16. Januar verschiedene Visualisierungstechniken dar. Aus der Bewertung durch Andreas Winter ergaben sich folgende neue / veränderte Anforderungen:

1. Die Web-Suche soll als Batch durchgeführt werden. Dieser ist nicht unterbrechbar.
2. Plagiate sollen farblich gekennzeichnet werden. Unterschiedliche Plagiatstypen erhalten eine unterschiedliche farbliche Kennzeichnung.
3. Zur entsprechenden Textpassage soll es einen Verweis auf die plagiierte Quelle geben.
4. Der Benutzer soll gefundene Quellen abwählen können.
5. Es sollen 3 Torten-Diagramme dargestellt werden: Wie sicher ist das gefundene ein Plagiat? Verhältnis von Plagiaten zum Gesamttext? Wie häufig wurden bestimmte Quellen verwendet?
6. Eine Übersicht soll den Status der Überprüfung / des Prozesses darstellen. Hier soll bei Abschluss der Überprüfung auch eine Kurz-Zusammenfassung dargestellt werden.
7. Es müssen eine Beschreibung der Visualisierung sowie eine Zusammenfassung über die Resultate textuell angezeigt werden.
8. Ergebnisdokumente sollen als PDF exportierbar sein.
9. Als Text-Codierung reicht UTF8.
10. Für die Web-Suche sollen keine kostenpflichtigen Dienste verwendet werden.